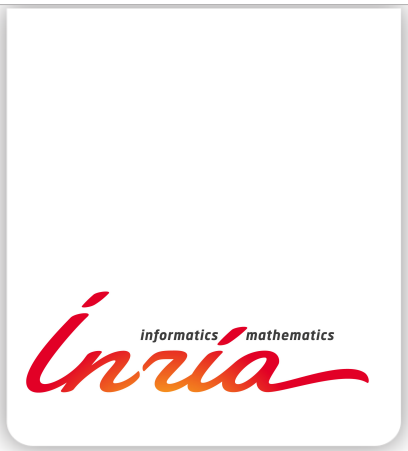
Low rank approximation of a sparse matrix based on LU factorization with column and row tournament pivoting

Laura Grigori, Sebastien Cayrols, James W. Demmel

RESEARCH REPORT
N° 8910
May 2016
Project-Team Alpines

ISSN 0249-6399 ISRN INRIA/RR--8910--FR+ENG



Low rank approximation of a sparse matrix based on LU factorization with column and row tournament pivoting

Laura Grigori*, Sebastien Cayrols†, James W. Demmel‡

Project-Team Alpines

Research Report n° 8910 — May 2016 — 35 pages

Abstract: In this paper we present an algorithm for computing a low rank approximation of a sparse matrix based on a truncated LU factorization with column and row permutations. We present various approaches for determining the column and row permutations that show a trade-off between speed versus deterministic/probabilistic accuracy. We show that if the permutations are chosen by using tournament pivoting based on QR factorization, then the obtained truncated LU factorization with column/row tournament pivoting, LU_CRTP, satisfies bounds on the singular values which have similarities with the ones obtained by a communication avoiding rank revealing QR factorization. Experiments on challenging matrices show that LU_CRTP provides a good low rank approximation of the input matrix and it is less expensive than the rank revealing QR factorization in terms of computational and memory usage costs, while also minimizing the communication cost. We also compare the computational complexity of our algorithm with randomized algorithms and show that for sparse matrices and high enough but still modest accuracies, our approach is faster.

Key-words: Rank revealing, LU and QR factorizations, column pivoting, minimize communication

* Inria Paris, Alpines, and UPMC Univ Paris 06, CNRS UMR 7598, Laboratoire Jacques-Louis Lions, Paris, France (laura.grigori@inria.fr. The work of this author is funded by the NLAFET project as part of European Union's Horizon 2020 research and innovation programme under grant agreement No 671633).

† INRIA Rocquencourt, Alpines, and UPMC Univ Paris 06, CNRS UMR 7598, Laboratoire Jacques-Louis Lions, Paris, France (sebastien.cayrols@inria.fr)

‡ Computer Science Division and Mathematics Department, UC Berkeley, CA 94720-1776, USA (demmel@cs.berkeley.edu).

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Approximation de rang faible pour les matrices creuses

Résumé : Ce papier introduit un algorithme pour calculer une approximation de rang faible d'une matrice creuse. Cet algorithme est basé sur une factorisation LU avec des permutations de lignes et de colonnes.

Mots-clés : calcul de rang, factorisations LU et QR, pivotage par colonnes, minimiser les communications

1 Introduction

In this paper we address the problem of computing a low rank approximation of a large sparse matrix by using a rank revealing LU factorization. This problem has numerous and diverse applications ranging from scientific computing problems such as fast solvers for integral equations to data analytics problems such as principal component analysis (PCA) or image processing. The singular value decomposition produces the best rank-k approximation, however it is expensive to compute. There are a number of less expensive approaches in the literature that approximate the singular value decomposition of a matrix, such as rank revealing QR and LU factorizations, or the Lanczos algorithm (see e.g. [8, 34]). In the recent years, several randomized algorithms have been introduced for this problem that aim at further decreasing the computational cost while obtaining accurate results with high probability. For recent surveys, see e.g. [24, 31]. While in this paper we discuss their usage to compute low rank approximations, there are many other important problems that require estimating the singular values of a matrix or its rank such as regularization, subset selection, and nonsymmetric eigenproblems (a more detailed description can be found in [5]).

In this paper we focus on sparse LU factorizations that are effective in revealing the singular values of a matrix, in terms of both accuracy and speed. For sparse matrices, direct methods of factorization lead to factors that are denser than the original matrix A. Since the R factor obtained from the QR factorization is the Cholesky factor of A^TA, it is expected that the factors obtained from a QR factorization are denser than the factors obtained from an LU factorization. Indeed, our focus is on obtaining a factorization that is less expensive than the rank revealing QR factorization in terms of computational and memory usage costs, while also minimizing the communication cost. The communication cost is one of the dominant factors for the performance of an algorithm on current and future computers [14], and classic algorithms based on row and/or column permutations are sub-optimal in terms of communication.

We begin by defining rank-revealing factorizations, and surveying prior work, in order to compare our work to it. Consider first the QR factorization with column permutations of a matrix $A \in \mathbb{R}^{m\times n}$ of the form

$$AP_c = QR = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}, \tag{1}$$

where $Q \in \mathbb{R}^{m\times m}$ is orthogonal, $R_{11} \in \mathbb{R}^{k\times k}$ is upper triangular, $R_{12} \in \mathbb{R}^{k\times(n-k)}$, and $R_{22} \in \mathbb{R}^{(m-k)\times(n-k)}$. We say that this is a rank revealing factorization (RRQR) if the column permutation matrix P_c is chosen such that

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})}, \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq q(k,n), \tag{2}$$

for any $1 \leq i \leq k$ and $1 \leq j \leq \min(m,n)-k$, where $q(k,n)$ is a low degree polynomial in n and k, and $\sigma_1(A) \geq \ldots \geq \sigma_n(A)$ are the singular values of A. In other words, the singular values of R_{11} approximate well the largest k singular values of A, while the singular values of R_{22} approximate well the $min(m,n)-k$ smallest singular values of A. Without loss of generality, here and in the rest of the paper we assume that the singular values of A and R are all nonzero. We present in more detail this factorization in section 2, as well as the *strong RRQR factorization* introduced in [23]. For a given k and a parameter $f > 1$, the results in [23] show that there exists a permutation P_c such that the factorization in equation (1) satisfies not only the bounds on singular values from inequality (2), but also bounds the absolute values of the elements of $R_{11}^{-1}R_{12}$. The RRQR factorization was introduced in [20] and the first algorithm to compute it was introduced in [4]. It is still one of the most used algorithms nowadays, even though it sometimes fails to

satisfy (2), for example on the so-called Kahan matrix [27]. It also only guarantees that the absolute value of the entries of $R_{11}^{-1}R_{12}$ is bounded by $O(2^n)$. We refer to this algorithm as QRCP, which is short for QR with Column Pivoting. When these algorithms are executed in a distributed memory environment, the matrix A is typically distributed over P processors by using a one-dimensional or two-dimensional (block) cyclic partitioning. Finding the column of maximum norm at each step of the factorization as in QRCP requires a reduction operation among processors, which costs $O(\log P)$ messages. QRCP exchanges $O(k \log P)$ messages for computing a rank-k approximation, and if the factorization proceeds until the end, it exchanges $O(n \log P)$ messages. A lower bound on the number of messages required for computing the QR factorization of a dense $n \times n$ matrix A (under certain hypotheses and when the memory size per processor is $O(n^2/P)$) [2] is $\Omega(\sqrt{P})$. Hence QRCP is not optimal in terms of the number of messages exchanged. A communication avoiding RRQR factorization, referred to as CARRQR, was introduced in [12]. This factorization selects k linearly independent columns by using tournament pivoting which requires only $O(\log P)$ messages. CARRQR is optimal in terms of communication, modulo polylogarithmic factors, on both sequential machines with two levels of slow and fast memory and parallel machines with one level of parallelism, while performing three times more floating point operations than QRCP. Extensive numerical experiments presented in [12] show that tournament pivoting reveals the singular values of a matrix with an accuracy close to the one obtained by QRCP. We will discuss it in more detail in section 2.

In this paper we focus on computing a low rank approximation of a matrix A and we consider different possible cases in which either the desired rank k is known, all singular values larger than a tolerance τ need to be approximated, or a gap in the singular values needs to be identified. We introduce a truncated LU factorization with column and row permutations which, for a given rank k, has the form

$$P_r A P_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{A}_{21}\bar{A}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix}, \tag{3}$$

$$S(\bar{A}_{11}) = \bar{A}_{22} - \bar{A}_{21}\bar{A}_{11}^{-1}\bar{A}_{12}, \tag{4}$$

where $A \in \mathbb{R}^{m\times n}$, $\bar{A}_{11} \in \mathbb{R}^{k,k}$, $\bar{A}_{22} \in \mathbb{R}^{m-k,n-k}$, and the rank-k approximation matrix $\tilde{A}_k$ is

$$\tilde{A}_k = \begin{pmatrix} I \\ \bar{A}_{21}\bar{A}_{11}^{-1} \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \end{pmatrix} = \begin{pmatrix} \bar{A}_{11} \\ \bar{A}_{21} \end{pmatrix} \bar{A}_{11}^{-1} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \end{pmatrix}. \tag{5}$$

The column and row permutations are chosen such that the singular values of $\bar{A}_{11}$ approximate the first k singular values of A, while the singular values of $S(\bar{A}_{11})$ approximate the last $min(m,n) - k$ singular values of A. For this, the factorization first selects k "most linearly independent" columns from the matrix A, permutes them to the leading positions, and then selects k "most linearly independent" rows from these columns. Depending on the methods used to select columns and rows, different algorithms can be obtained, with different bounds on the revealed singular values and on the numerical stability of the truncated LU factorization. The design space for selecting the k columns and rows can be summarized by the following list (other possibilities have been proposed, e.g. choosing the k-by-k submatrix of nearly maximum determinant [33, 32]):

1. Select k linearly independent columns of A (call result B), by using
   (a) (strong) QRCP / tournament pivoting using QR,
   (b) LU / tournament pivoting based on LU, with some form of pivoting (column, complete, rook),

(c) randomization: premultiply $X = ZA$ where random matrix Z is short and wide, then pick k rows from X^T, by some method from 2) below,

(d) tournament pivoting based on randomized algorithms to select columns at each step.

2. Select k linearly independent rows of B, by using

(a) (strong) QRCP / tournament pivoting based on QR on B^T, or on Q^T, the rows of the thin Q factor of B,

(b) LU / tournament pivoting based on LU with some form of pivoting (row, TSLU [22], complete, rook) on B,

(c) tournament pivoting based on randomized algorithms to select rows, or other suitable randomized algorithms.

These various approaches show a trade-off between speed versus deterministic / probabilistic accuracy and are presented in order of expected decreasing accuracy. Concerning speed, the selections based on QR are more expensive than those based on LU, in terms of floating point operations and also memory usage, however they are expected to provide more accurate approximations of the singular values. The classic pivoting strategies used in LU and QR factorizations are sub-optimal in terms of communication, while tournament pivoting provides a communication optimal alternative, with worse theoretical bounds on the approximations of the singular values.

The second formulation of $\tilde{A}_k$ from (5) is called a CUR decomposition (see [21, 35, 31] and references therein), a popular low rank approximation factorization in data analytics in which C and R are columns and rows selected from the matrix A, and $\bar{A}_{11}^{-1}$, and hence very sparse. In many cases of interest, $\tilde{A}_k$ is applied to a vector, and $\bar{A}_{11}^{-1}$ should not be computed, but instead its LU or QR factorization should be used. The column/row selection design space described above can be used to obtain such a CUR decomposition. The randomized algorithms reviewed in [31] generally fit in category (1c) of our design space.

In this design space, we show that LU_CRTP, a factorization that uses tournament pivoting based on QR from part 1a) above and tournament pivoting based on QR on Q^T from part 2a) above, is a good choice in terms of both speed and accuracy. We discuss it in more detail in section 3 and show that LU_CRTP allows us not only to obtain bounds on the approximated singular values, the stability of the LU factorization, but also to minimize communication. The obtained factorization satisfies the following properties

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(\bar{A}_{11})}, \frac{\sigma_j(S(\bar{A}_{11}))}{\sigma_{k+j}(A)} \leq q(m,n,k), \tag{6}$$

$$\rho_l(\bar{A}_{21}\bar{A}_{11}^{-1}) \leq F_{TP} \tag{7}$$

for any $1 \leq l \leq m-k$, $1 \leq i \leq k$, and $1 \leq j \leq \min(m,n)-k$, where $\rho_l(B)$ denotes the 2-norm of the l-th row of B, and F_{TP} is a quantity arising in the bound on singular values obtained after column tournament pivoting based on QR [12]. If a binary tree is used during tournament pivoting to select k columns from n, then $F_{TP} \leq \frac{1}{\sqrt{2k}}(n/k)^{\log_2(\sqrt{2}fk)}$, and $q(m,n,k) = \sqrt{(1+F_{TP}^2(n-k))(1+F_{TP}^2(m-k))}$, where f is a small constant related to the strong RRQR factorization used during tournament (typically $f=2$). The bound on F_{TP} can be regarded as a polynomial in n for a fixed k and f. For more details see Theorem 3.1. The second inequality (7) is important for bounding element growth in the LU factorization which governs its numerical stability. All these results are obtained assuming infinite precision, and

they are expected not to hold when the singular values approach machine precision, and so may be significantly changed by roundoff error.

The existence of a rank revealing LU factorization has been proven by Pan in [33], who shows that there are permutation matrices P_r, P_c such that the factorization from (3) satisfies

$$1 \leq \frac{\sigma_k(A)}{\sigma_{min}(\bar{A}_{11})}, \frac{\sigma_{max}(S(\bar{A}_{11}))}{\sigma_{k+1}(A)} \leq k(n-k)+1. \quad (8)$$

The existence of a stronger LU factorization has been proven by Miranian and Gu in [32], which in addition to (6) and (7) also upper bounds $||\bar{A}_{11}^{-1}\bar{A}_{12}||_{max}$ by a low degree polynomial in k, n and m. Our bounds from (6) are slightly worse than those from (8) and also slightly worse than those obtained by CARRQR for which $q(m,n,k) = \sqrt{1+F_{TP}^2(n-k)}$ (see Theorem 2.3 for more details). A better bound than (7) can be obtained by using strong QRCP for $2a$) above, in which case $||\bar{A}_{21}\bar{A}_{11}^{-1}||_{max} \leq f$, similar to the LU factorization with panel rank revealing pivoting from [29]. All those better bounds are obtained by algorithms that require more computation and/or more communication, while our algorithm provides a good trade-off between speed and accuracy.

Once the first k singular values of A are approximated, it might be the case that subsequent singular values need to be computed. One example is when the algorithm is used to compute a low rank approximation and no gap has been identified in the singular values. Then the block LU factorization algorithm continues recursively on $S(\bar{A}_{11})$ until a rank K (which is a multiple of k) has been determined. Some of the previous bounds become exponential, where the exponent is the number of steps of tournament pivoting, K/k, for more details see Theorem 3.3. But in practice the algorithm is effective in revealing the spectrum of A and provides a good low rank approximation.

In section 4 we discuss the cost of our truncated LU factorization for the case when the rank k is known, first for matrices with arbitrary sparsity structure and then for matrices with small separators. A sparse LU factorization with partial pivoting or a sparse QR factorization would first reorder the columns of the matrix to reduce fill (based on the structure of A^TA in the case of the QR factorization), and then would perform the LU or QR factorization. In the case of the QR factorization, the sparsity pattern of the R factor does not depend on row permutations, but depends on column permutations. When the factorization needs to be performed with column pivoting, since the column pivoting depends on the numerical values of the matrix A, reordering the columns of A before the factorization is not useful for the classic QRCP factorization. However, this order is important for tournament pivoting, since it can reduce the fill-in during the QR factorization of subsets of columns. For matrices with arbitrary sparsity structure, an upper bound on the number of flops required by one tournament pivoting based on QR factorization to choose k columns, executed sequentially, which dominates the cost of our algorithm, is $O(k^2 nnz(A))$. When the algorithm is executed in parallel on P processors, an upper bound on the number of flops is $O(k^2 \frac{nnz(A)}{P} \log \frac{n}{k})$ and the number of messages exchanged is $O(\log P)$. Given the assumptions we make to derive the bounds on the number of flops, we expect that these bounds are pessimistic.

For comparison, one of the most efficient randomized algorithms for computing a low rank approximation of a sparse matrix is due to Clarkson and Woodruff [7]. For an $n \times n$ matrix A, it computes a rank-k approximation which satisfies $||A - LDW^T||_F \leq (1+\epsilon)||A - A_k||_F$, where L and W are of dimension $n \times k$, D is of dimension $k \times k$, and A_k is the best rank-k approximation. This algorithm requires $O(nnz(A)) + \bar{O}(nk^2\epsilon^{-4} + k^3\epsilon^{-5})$ flops, where $\bar{O}(f) = f \cdot \log^{O(1)}(f)$. By ignoring the constants in the asymptotic $O()$ notation, if ϵ is chosen such that $nnz(A) \leq \bar{O}(n\epsilon^{-4})$, or after rearranging, $\epsilon \leq \frac{1}{(\frac{nnz(A)}{n})^{1/4}}$, our algorithm is faster than the randomized approach, while also being deterministic. For example, if $\epsilon = 0.5$ or $\epsilon = 0.1$, then our algorithm is faster if the average number of nonzeros per column of A is smaller than 16 or 10^4 respectively. We note

that even for a fixed ϵ, comparing the theoretical bounds on accuracy of the two algorithms is difficult, since we bound the error of the approximated singular values, while Clarkson and Woodruff bound the Frobenius norm of the error of the low rank approximation. A meaningful comparison of the accuracy of the two algorithms should include varying ϵ and comparing the results on matrices from different applications. This remains future work.

In section 5 we present experimental results which compare LU_CRTP with results obtained by using the singular value decomposition and QRCP. On a set of selected matrices that were used in previous papers on rank revealing factorizations, we show that our factorization algorithm approximates well the singular values of the matrix A. The ratio of the absolute values of the singular values of the block A_{11} to the corresponding singular values of A is at most 13 (except for a difficult matrix, the devil's stairs, for which this ratio is 27). We also compare the number of nonzeros in the low rank approximation obtained by our algorithm with respect to a low rank approximation obtained by using QRCP or LU with partial pivoting. For the same rank, we obtain a factor of up to 208x fewer nonzeros with respect to QRCP, and in some cases fewer nonzeros than LU with partial pivoting.

2 Background

In this section we review some of the factorizations and their properties that will be used in our rank revealing LU factorization. We focus in particular on rank revealing and strong rank revealing QR and LU factorizations.

2.1 Notation

We use Matlab notation. Given two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times k}$, the matrix $C \in \mathbb{R}^{m\times(n+k)}$ obtained by putting the two matrices next to each other is referred to as $C = [A,\ B]$. Given two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{k\times n}$, the matrix $C \in \mathbb{R}^{(m+k)\times n}$ obtained by putting A on top of B is referred to as $C = [A;\ B]$. The submatrix of A formed by rows i to j and columns l to k is referred to as $A(i:j, l:k)$. The element in position (i, j) of A is referred to as $A(i, j)$. Very often, when a matrix is partitioned into a block matrix, we give the dimensions of a few blocks and we assume the reader can deduce the dimensions of the remaining blocks. Given a matrix A partitioned as $A = [A_{00}, \ldots, A_{T,0}]$, we refer to the sub-matrices A_{00} to $A_{T,0}$ as $A_{0:T,0}$. Similar notation is used for a block matrix with multiple rows.

The absolute values of the matrix A are referred to as $|A|$. The max norm is defined as $||A||_{max} = \max_{i,j} |A_{ij}|$. We refer to the 2-norm of the j-th row of A as $\rho_j(A)$, the 2-norm of the j-th column of A as $\chi_j(A)$, and the 2-norm of the j-th row of A^{-1} as $\omega_j(A)$.

2.2 Rank revealing QR factorizations

Consider first the QR factorization with column permutations of a matrix $A \in \mathbb{R}^{m\times n}$ of the form

$$AP_c = QR = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}, \tag{9}$$

where $Q \in \mathbb{R}^{m\times m}$ is orthogonal, $R_{11} \in \mathbb{R}^{k\times k}$ is upper triangular, $R_{12} \in \mathbb{R}^{k\times(n-k)}$, and $R_{22} \in \mathbb{R}^{(m-k)\times(n-k)}$. We say that this factorization is *rank revealing* if

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})}, \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq q(k, n), \tag{10}$$

for any $1 \leq i \leq k$ and $1 \leq j \leq \min(m,n)-k$, where $q(k,n)$ is a low degree polynomial in k and n. Note that definitions of a rank revealing factorization as given in [26, 6] bound only $\sigma_{max}(R_{11})$ and $\sigma_{min}(R_{22})$ with respect to the singular values of A, so that our definition is stricter. The two following theorems recall the properties of a strong rank revealing QR factorization [23].

Theorem 2.1 *(Gu and Eisenstat [23]) Let A be an $m \times n$ matrix and let $1 \leq k \leq \min(m,n)$. For any given parameter $f > 1$, there exists a permutation P_c such that*

$$AP_c = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}, \tag{11}$$

where R_{11} is $k \times k$ and

$$\left(R_{11}^{-1} R_{12}\right)_{i,j}^2 + \omega_i^2 \left(R_{11}\right) \chi_j^2 \left(R_{22}\right) \leq f^2, \tag{12}$$

for any $1 \leq i \leq k$ and $1 \leq j \leq n-k$.

The inequality (12) defining a strong rank revealing factorization allows to bound the singular values of R_{11} and R_{22} as in a rank revealing factorization, while also upper bounding the absolute values of $R_{11}^{-1} R_{12}$.

Theorem 2.2 *(Gu and Eisenstat [23]) Let the factorization in Theorem 2.1 satisfy inequality (12). Then*

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})}, \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq \sqrt{1 + f^2 k(n-k)},$$

for any $1 \leq i \leq k$ and $1 \leq j \leq \min(m,n) - k$, .

The communication avoiding RRQR factorization, referred to as CARRQR [12], computes a rank revealing factorization by using a block algorithm that selects k columns at each iteration, permutes them to be the leading columns, computes k steps of a QR factorization with no pivoting, and then proceeds recursively on the trailing matrix. The k columns are selected by using tournament pivoting as described later in this section. It is shown in [12] that CARRQR computes a factorization as in equation (11) and a permutation P_c that satisfies

$$\chi_j^2 \left(R_{11}^{-1} R_{12}\right) + \left(\chi_j \left(R_{22}\right) / \sigma_{\min}(R_{11})\right)^2 \leq F_{TP}^2, \text{ for } j = 1, \ldots, n-k. \tag{13}$$

Here F_{TP} depends on k, f, n, the shape of reduction tree used during tournament pivoting, and the number of iterations of CARRQR. The following theorem, which is a relaxed version of Theorem 2.1, shows that CARRQR reveals the rank in a similar way to strong RRQR factorization, for more details see [12].

Theorem 2.3 *Assume that there exists a permutation P_c for which the QR factorization*

$$AP_c = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}, \tag{14}$$

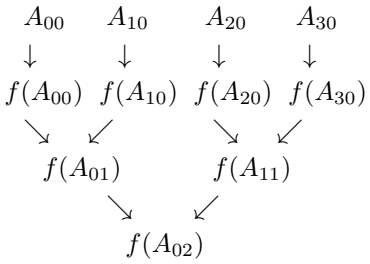
where R_{11} is $k \times k$ and satisfies (13)*. Then*

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})}, \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq \sqrt{1 + F_{TP}^2 (n-k)}, \tag{15}$$

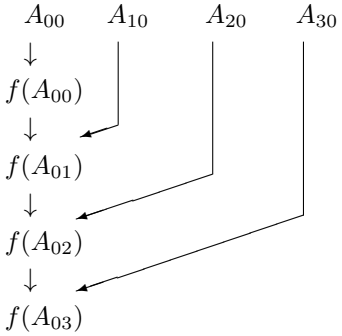
for any $1 \leq i \leq k$ and $1 \leq j \leq \min(m,n) - k$.

In this work we use only one step of tournament pivoting, which corresponds to one iteration of CARRQR. We refer to this factorization as QR_TP(A,k) and Algorithm 1 details its computation. A more detailed description of how this algorithm can be implemented in a parallel environment can be found in [12]. Tournament pivoting selects k columns from the matrix A by using a reduction tree. It first divides the matrix A into $n/(2k)$ subsets of columns and at the leaves of the reduction tree, it selects from each subset k columns by using (strong) rank revealing QR factorization. At each node of the subsequent levels of the reduction tree, a new matrix is formed by adjoining next to each other the two subsets of candidate columns selected by its children. A new set of k candidate columns is selected by using (strong) rank revealing QR factorization. The columns selected at the root of the reduction tree are the final columns selected by tournament pivoting. The factorization from Theorem 2.3, which satisfies inequality (13), is obtained by permuting those columns to the first k positions. We note that in our experimental results as well as in those reported in [12], we use QRCP for the selection of k columns at each node of the reduction tree. However the bounds on F_{TP} are obtained by using strong rank revealing QR factorization.

We present in the following picture a binary tree based tournament pivoting that selects k columns from A. In this example, the matrix A is partitioned into 4 subsets of columns, $A = [A_{00}, A_{10}, A_{20}, A_{30}]$. At the leaves of the reduction tree, for each subset of columns A_{0j}, $f(A_{0j})$ selects k columns by using strong rank revealing QR factorization of A_{0j}. Then at each node of the reduction tree, a new matrix A_{ij} is obtained by adjoining the columns selected by the children of the node, and $f(A_{ij})$ selects k columns by using strong rank revealing QR factorization of A_{ij}.



The flat tree based QR with tournament pivoting is presented in the following picture.



Corollary 2.4 *(Corollaries 2.6 and 2.7 from [12]) The selection of k columns of the $m \times n$ matrix A using QR with binary tree based tournament pivoting reveals the rank of A in the sense*

Algorithm 1 QR_TP (A,k): Select k linearly independent columns from a matrix A by using QR factorization with binary tree based tournament pivoting

Input $A \in \mathbb{R}^{m\times n}$, number of columns to select k
Output P_c such that $(AP_c)(:, 1:k)$ are the k selected columns
Partition the matrix $A = [A_{00}, \ldots, A_{n/k,0}]$, where $A_{i0} \in \mathbb{R}^{m\times 2k}$, $i = 1, \ldots n/(2k)$ // Assume n is a multiple of $2k$
for each level in the reduction tree $j = 0$ to $\log_2 n/(2k) - 1$ **do**
  for each node i in the current level j **do**
    if $j = 0$ (at the leaves of the reduction tree) **then**
      A_{i0} is the i-th block of $2k$ columns of A
    else Form A_{ij} by putting next to each other the two sets of k column candidates selected by the children of node j
    end if
    Select k column candidates by computing $A_{ij} = Q_1 R_1$ and then computing a RRQR factorization of R_1, $R_1 P_{c_2} = Q_2 \begin{pmatrix} R_2 & * \\ & * \end{pmatrix}$
    if j is the root of the reduction tree **then**
      Return P_c such that $(AP_c)(:, 1:k) = (A_{ij}P_{c_2})(:, 1:k)$
    else Pass the k selected columns, $AP_{c_2}(:, 1:k)$ to the parent of i
    end if
  end for
end for

of Theorem 2.3, with bound

$$F_{TP-BT} \leq \frac{1}{\sqrt{2k}} \left(\sqrt{2} f k\right)^{\log_2(n/k)} = \frac{1}{\sqrt{2k}} (n/k)^{\log_2\left(\sqrt{2} f k\right)} \tag{16}$$

If tournament pivoting uses a flat tree, then the bound becomes

$$F_{TP-FT} \leq \frac{1}{\sqrt{2k}} \left(\sqrt{2} f k\right)^{n/k}. \tag{17}$$

In the case of a binary tree, the bound in (16) can be regarded as a polynomial in n in general for a fixed k and f. The bound (17) obtained for a flat tree is exponential in n, however is a rapidly decreasing function of k. Even if this suggests that the singular values approximated by the factorization might be far from the singular values of A, the extensive numerical experiments performed in [12] show that both binary tree and flat tree are effective in approximating the singular values of A.

2.3 Orthogonal matrices

Consider an orthogonal matrix $Q \in \mathbb{R}^{m\times m}$ and its partitioning as

$$Q = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \tag{18}$$

where $Q_{11} \in \mathbb{R}^{k\times k}$. The QR_TP factorization of $(Q_{11}; Q_{21})^T$ leads to the following factorization

$$P_r Q = \begin{pmatrix} \bar{Q}_{11} & \bar{Q}_{12} \\ \bar{Q}_{21} & \bar{Q}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{Q}_{21}\bar{Q}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{Q}_{11} & \bar{Q}_{12} \\ & S(\bar{Q}_{11}) \end{pmatrix} \tag{19}$$

where $S(\bar{Q}_{11}) = \bar{Q}_{22} - \bar{Q}_{21}\bar{Q}_{11}^{-1}\bar{Q}_{12} = \bar{Q}_{22}^{-T}$ (see [33], proof of Theorem 3.7). This factorization satisfies the following bounds

$$\rho_j(\bar{Q}_{21}\bar{Q}_{11}^{-1}) \leq F_{TP}, \tag{20}$$

$$\frac{1}{q_2(k,m)} \leq \sigma_i(\bar{Q}_{11}) \leq 1, \tag{21}$$

for all $1 \leq i \leq k$, $1 \leq j \leq m-k$, where $q_2(k,m) = \sqrt{1 + F_{TP}^2(m-k)}$. F_{TP} is the bound obtained from QR with tournament pivoting, given in equation (16) for a binary tree and in equation (17) for a flat tree.

The singular values of $\bar{Q}_{11}$ are also bounded in Lemma 3.4 from [33]. In that paper, $vol(\bar{Q}_{11})$ is the absolute value of the determinant of $\bar{Q}_{11}$. It is a local μ-maximum in Q if for all neighboring sub-matrices Q' which differ from $\bar{Q}_{11}$ in exactly one column or one row, $\mu \cdot vol(\bar{Q}_{11}) \geq vol(Q')$, for some $\mu \geq 1$. The permutation P_r is chosen in [33] such that $vol(\bar{Q}_{11})$ is a local μ-maximum in Q and $q_2(k,m) = \sqrt{1 + k(m-k)\mu^2}$.

By using the properties of the CS decomposition of an orthogonal matrix, we also have that $\sigma_{min}(\bar{Q}_{11}) = \sigma_{min}(\bar{Q}_{22})$.

3 LU_CRTP: Block LU factorization with column/row tournament pivoting

In this section we describe the algebra of our low rank approximation factorization based on LU decomposition with column/row tournament pivoting. There are two different cases that we consider here, in the first the rank k of the approximation is known, in the second the rank K of the approximation is to be determined while computing the factorization. We also discuss the numerical properties of our LU factorization. We present bounds on the singular values of the obtained factors with respect to the original matrix A. In addition we also discuss the backward stability of the LU factorization.

3.1 Rank-k approximation, when k is known

We consider first the case in which the rank k is known. Given a matrix $A \in \mathbb{R}^{m \times n}$, we are seeking a factorization of the form

$$\bar{A} = P_r A P_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{A}_{21}\bar{A}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix} \tag{22}$$

where

$$S(\bar{A}_{11}) = \bar{A}_{22} - \bar{A}_{21}\bar{A}_{11}^{-1}\bar{A}_{12}. \tag{23}$$

The permutation matrices P_r, P_c are chosen such that $\bar{A}_{11}$ approximates well the largest k singular values of A, while $\bar{A}_{22}$ approximates well the smallest $n-k$ singular values. If this factorization is run to completion, that is if $S(\bar{A}_{11})$ is factored by using the same procedure and the factorization obtained is $P_r' A P_c' = LU$, then it is known that the stability of the factorization depends on the ratio $||\hat{L}||_{max}||\hat{U}||_{max}/||A||_{max}$, where $\hat{L}$ and $\hat{U}$ are the computed factors. For more details see [15], and also [29]. We make the assumption that $||\hat{L}||_{max}||\hat{U}||_{max} \approx ||L||_{max}||U||_{max}$, i.e. that roundoff does not change the norms of the L and U factors very much, and in the following we are interested in bounding $||L||_{max}$ and $||U||_{max}$. For this the elements of $|\bar{A}_{21}\bar{A}_{11}^{-1}|$ need to be bounded.

The algorithm selects the first k columns by using QR factorization with tournament pivoting on the matrix A (see Algorithm 1),

$$AP_c = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix} = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}$$

where $Q \in \mathbb{R}^{m \times m}$, $Q_{11}, R_{11} \in \mathbb{R}^{k \times k}$. As discussed in section 2.2, R_{11} reveals the largest k singular values of A, while R_{22} reveals the smallest $n-k$ singular values of A. However, with respect to the factorization in (22), we want $\bar{A}_{11}$ to reveal the largest k singular values of A and $S(\bar{A}_{11})$ the smallest $n-k$ singular values of A. For this, we select k rows from the first k columns of Q by using QR with tournament pivoting on $Q(:, 1:k)^T$ and obtain the factorization,

$$P_r Q = \begin{pmatrix} \bar{Q}_{11} & \bar{Q}_{12} \\ \bar{Q}_{21} & \bar{Q}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{Q}_{21}\bar{Q}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{Q}_{11} & \bar{Q}_{12} \\ & S(\bar{Q}_{11}) \end{pmatrix} \tag{24}$$

where

$$S(\bar{Q}_{11}) = \bar{Q}_{22} - \bar{Q}_{21}\bar{Q}_{11}^{-1}\bar{Q}_{12} = \bar{Q}_{22}^{-T}. \tag{25}$$

such that $\rho_j(\bar{Q}_{21}\bar{Q}_{11}^{-1}) = \rho_j(\bar{A}_{21}\bar{A}_{11}^{-1}) \leq F_{TP}$, for all $1 \leq j \leq m-k$, is upper bounded as in equation (20), and the singular values of $\bar{Q}_{11}$ and $\bar{Q}_{22}$ are bounded as in equation (21) (see section 2.3). This will allow us to show that $\bar{A}_{11}$ and $S(\bar{A}_{11})$ reveal the singular values of A.

By applying the column and row permutations on the matrix A we obtain the desired factorization,

$$\begin{aligned} P_r A P_c &= \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{Q}_{21}\bar{Q}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{Q}_{11} & \bar{Q}_{12} \\ & S(\bar{Q}_{11}) \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix} \\ &= \begin{pmatrix} I & \\ \bar{A}_{21}\bar{A}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix} \end{aligned} \tag{26}$$

where

$$\begin{aligned} \bar{Q}_{21}\bar{Q}_{11}^{-1} &= \bar{A}_{21}\bar{A}_{11}^{-1}, \\ S(\bar{A}_{11}) &= S(\bar{Q}_{11})R_{22} = \bar{Q}_{22}^{-T}R_{22}. \end{aligned}$$

A similar derivation has been used by Pan in [33] to prove the existence of a rank revealing LU factorization. However, this derivation was not used in the rank revealing algorithms introduced in that paper. Indeed computing P_c by using a classic algorithm like QRCP would require computing R_{22}, and hence R_{22} could be used in the subsequent steps instead of computing an LU factorization and its Schur complement $S(\bar{A}_{11})$. In addition, the derivation in [33] does not bound $||A_{21}A_{11}^{-1}||_{max}$, which is important for the numerical stability of the LU factorization.

We present the LU factorization with column/row tournament pivoting obtained by the above procedure, which we refer to as LU_CRTP (A, k) in Algorithm 2. The first step of the algorithm (line 3) selects k columns by using QR with tournament pivoting on A. This factorization computes strong RRQR factorizations of subsets of $2k$ columns, but it never computes the complete QR factorization that was used to derive the algebra in equation (26). Once the QR factorization of the selected k columns from tournament pivoting is computed in step 4, k rows are selected by using tournament pivoting on $Q(:, 1:k)^T$ in step 5. The row and column permutations are applied to A and Q in step 6.

The computation of L_{21} in step 7 requires special attention. In infinite precision, $\bar{A}_{21}\bar{A}_{11}^{-1} = \bar{Q}_{21}\bar{Q}_{11}^{-1}$, however this might not be the case in finite precision. Due to round-off error, the computation of $\bar{Q}_{21}\bar{Q}_{11}^{-1}$ is numerically more stable than the computation of $\bar{A}_{21}\bar{A}_{11}^{-1}$. In the sparse

case, not only the numerical values differ, but also $\bar{Q}_{21}\bar{Q}_{11}^{-1}$ has more nonzeros than $\bar{A}_{21}\bar{A}_{11}^{-1}$. This is because $\bar{Q}_{11}$ and $\bar{Q}_{21}$ are denser than $\bar{A}_{11}$ and $\bar{A}_{21}$. The additional nonzeros correspond to exact cancellations and they are due to round-off errors. In our numerical experiments we observe that they have very small values. We do not investigate further this issue in this paper and we use a heuristic in which we first compute $L_{21} = \bar{A}_{21}\bar{A}_{11}^{-1}$ and we check if $||\bar{A}_{21}\bar{A}_{11}^{-1}||_{max}$ is small. In all our experimental results, this is indeed the case, and $||\bar{A}_{21}\bar{A}_{11}^{-1}||_{max}$ is at most 1.49. We do not exclude the possibility that $||\bar{A}_{21}\bar{A}_{11}^{-1}||_{max}$ can be large, in particular for nearly singular matrices. In this case, one should compute $\bar{Q}_{21}\bar{Q}_{11}^{-1}$ and possibly drop the elements smaller than a certain threshold. We again do not investigate this aspect further in the paper. We note that Algorithm 1 selects the columns and the rows that can be used in a CUR decomposition, in which case step 7 can be omitted.

Algorithm 2 LU_CRTP(A, k): rank-k truncated LU factorization with column/row tournament pivoting of a matrix A

1: **Input** $A \in \mathbb{R}^{m\times n}$, target rank k

2: **Output** permutation matrices P_r, P_c, rank-k truncated factorization L_kU_k, factor R_k, such that $(AP_c)(1:k) = Q_kR_k$,

$$\begin{aligned} P_rAP_c &= \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{A}_{21}\bar{A}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix}, \\ L_k &= \begin{pmatrix} I \\ \bar{A}_{21}\bar{A}_{11}^{-1} \end{pmatrix} = \begin{pmatrix} I \\ L_{21} \end{pmatrix}, \; U_k = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \end{pmatrix}, \end{aligned}$$

where $L_k \in \mathbb{R}^{m\times k}$, $U_k \in \mathbb{R}^{k\times n}$, $R_k \in \mathbb{R}^{k\times k}$, and the remaining matrices have corresponding dimensions.
Note that $S(\bar{A}_{11})$ is not computed in the algorithm.

3: Select k columns by using QR with tournament pivoting on A, Algorithm 1,
$P_c \leftarrow QR\_TP(A, k)$

4: Compute the thin QR factorization of the selected columns,
$(AP_c)(:, 1:k) = Q_kR_k$, where $Q_k \in \mathbb{R}^{m\times k}$ and $R \in \mathbb{R}^{k\times k}$

5: Select k rows by using QR with tournament pivoting on Q_k^T,
$P_r \leftarrow QR\_TP(Q_k^T, k)$

6: Let
$$\bar{A} = P_rAP_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix}, \quad P_rQ_k = \begin{pmatrix} \bar{Q}_{11} \\ \bar{Q}_{21} \end{pmatrix}$$

7: Compute
$L_{21} = \bar{Q}_{21}\bar{Q}_{11}^{-1} = \bar{A}_{21}\bar{A}_{11}^{-1}$ (see discussion in the text)

In the following theorem we show that the LU_CRTP (A, k) factorization reveals the singular values of A, and in addition also bounds element growth in the LU factorization.

Theorem 3.1 *Let A be an $m \times n$ matrix. The LU_CRTP(A, k) factorization obtained by using Algorithm 2,*

$$\bar{A} = P_rAP_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} I & \\ \bar{Q}_{21}\bar{Q}_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ & S(\bar{A}_{11}) \end{pmatrix} \tag{27}$$

where

$$S(\bar{A}_{11}) = \bar{A}_{22} - \bar{A}_{21}\bar{A}_{11}^{-1}\bar{A}_{12} = \bar{A}_{22} - \bar{Q}_{21}\bar{Q}_{11}^{-1}\bar{A}_{12}, \tag{28}$$

satisfies the following properties

$$\rho_l(\bar{A}_{21}\bar{A}_{11}^{-1}) = \rho_l(\bar{Q}_{21}\bar{Q}_{11}^{-1}) \leq F_{TP}, \quad (29)$$

$$||S(\bar{A}_{11})||_{max} \leq \min\left((1+F_{TP}\sqrt{k})||A||_{max}, F_{TP}\sqrt{1+F_{TP}^2(m-k)}\sigma_k(A)\right) \quad (30)$$

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(\bar{A}_{11})}, \frac{\sigma_j(S(\bar{A}_{11}))}{\sigma_{k+j}(A)} \leq q(m,n,k), \quad (31)$$

for any $1 \leq l \leq m-k$, $1 \leq i \leq k$, *and* $1 \leq j \leq \min(m,n)-k$. *Here* F_{TP} *is the bound obtained from QR with tournament pivoting, as in Corollary 2.4, and* $q(m,n,k) = \sqrt{(1+F_{TP}^2(n-k))\,(1+F_{TP}^2(m-k))}$.

Proof 3.2 *The left part of equation* (31) *is satisfied for any permutation matrices* P_r, P_c *by the interlacing property of singular values. The factorization from equation* (27) *can be written as in equation* (26)*, where*

$$\bar{A}_{11} = \bar{Q}_{11}R_{11}, \quad (32)$$

$$S(\bar{A}_{11}) = S(\bar{Q}_{11})R_{22} = \bar{Q}_{22}^{-T}R_{22}. \quad (33)$$

The permutation P_c *and the* R *factor from equation* (26) *are obtained by using QR with tournament pivoting, and the singular values of* R_{11} *satisfy the bounds from equation* (15)*. The row permutation* P_r *and* $\bar{Q}_{11}$ *are obtained by using QR with tournament pivoting, and as discussed in section 2.3, the singular values of* $\bar{Q}_{11}$ *satisfy the bounds from equation* (21)*. We obtain*

$$\sigma_i(\bar{A}_{11}) \geq \sigma_{min}(\bar{Q}_{11})\sigma_i(R_{11}) \geq \frac{1}{q_1(n,k)q_2(m,k)}\sigma_i(A),$$

where $q_1(n,k) = \sqrt{1+F_{TP}^2(n-k)}$, $q_2(m,k) = \sqrt{1+F_{TP}^2(m-k)}$. *Note that* $\sigma_{min}(\bar{Q}_{11}) = \sigma_{min}(\bar{Q}_{22})$. *We also have that*

$$\sigma_j(S(\bar{A}_{11})) = \sigma_j(S(\bar{Q}_{11})R_{22}) \leq ||S(\bar{Q}_{11})||_2\sigma_j(R_{22}) \leq q_1(n,k)q_2(m,k)\sigma_{k+j}(A).$$

By letting $q(m,n,k) = q_1(n,k)q_2(m,k)$ *we obtain the bounds on the singular values in equation* (31).

To bound element growth in the L factor, equation (29)*, we note that* $\bar{A}_{21}\bar{A}_{11}^{-1} = \bar{Q}_{21}\bar{Q}_{11}^{-1}$. *As shown in section 2.3, we have that* $\rho_l(\bar{A}_{21}\bar{A}_{11}^{-1}) = \rho_l(\bar{Q}_{21}\bar{Q}_{11}^{-1}) \leq F_{TP}$, *for each row* l *of* $\bar{A}_{21}\bar{A}_{11}^{-1}$. *Element growth in* $S(\bar{A}_{11})$ *is bounded as follows.*

$$\begin{aligned} |S(\bar{A}_{11})(i,j)| &= |\bar{A}_{22}(i,j) - (\bar{A}_{21}\bar{A}_{11}^{-1})(i,:)\bar{A}_{12}(:,j)| \\ &\leq ||A||_{max} + ||(\bar{A}_{21}\bar{A}_{11}^{-1})(i,:)||_2||\bar{A}_{12}(:,j)||_2 \leq ||A||_{max} + \rho_i(\bar{A}_{21}\bar{A}_{11}^{-1})\sqrt{k}||A||_{max} \\ &\leq (1+F_{TP}\sqrt{k})||A||_{max} \end{aligned}$$

In addition, we use the fact that the QR factorization with tournament pivoting that selects k *columns satisfies equation* (13)*, and we obtain* $\chi_j(R_{22}) = ||R_{22}(:,j)||_2 \leq F_{TP}\sigma_{min}(R_{11}) \leq F_{TP}\sigma_k(A)$. *The absolute value of the element in position* (i,j) *of* $S(\bar{A}_{11})$ *can be bounded as follows,*

$$\begin{aligned} |S(\bar{A}_{11})(i,j)| &= |\bar{Q}_{22}^{-T}(i,:)R_{22}(:,j)| \leq ||\bar{Q}_{22}^{-1}(:,i)||_2||R_{22}(:,j)||_2 \\ &\leq ||\bar{Q}_{22}^{-1}||_2||R_{22}(:,j)||_2 = ||R_{22}(:,j)||_2/\sigma_{min}(\bar{Q}_{22}) \leq F_{TP}q_2(m,k)\sigma_k(A). \end{aligned}$$

3.2 Rank-K approximation, when K is to be determined

We consider now the case when the rank of the approximation needs to be determined during the factorization. Our factorization determines an overestimation K of the rank such that the approximated singular values are larger than a tolerance τ. The overestimation K is a multiple of k, i.e. $K = tk$ and the rank has a value between $(t-1)k$ and $K = tk$. Another option is to identify a gap in the singular values; we do not discuss this case here, however the algorithm that we present can be easily adapted.

Given an initial guess of the rank k, the factorization computes one step of a block LU factorization with column/row tournament pivoting, as described in the previous section and obtains the factorization from equation (22). If the approximated singular values are all larger or equal than τ, then the factorization continues recursively on the trailing matrix $S(\bar{A}_{11})$. We refer to this factorization as $LU\_CRTP(A, k, \tau)$, which is presented in Algorithm 3. After T steps, we obtain the factorization from equation (34). In the following theorem we give relations between the singular values of the diagonal blocks U_{ii}, $1 \leq i \leq T$ and the largest K singular values of A, and also between the singular values of $U_{T+1,T+1}$ and the smallest $n-K$ singular values of A. We observe that with respect to the results in Theorem 3.1, the bounds here become exponential, where the exponent is the number of steps of tournament pivoting, K/k. An exponentially growing bound cannot be avoided without additional permutations, since in the special case $k = 1$, the algorithm reduces to QRCP, where we know exponential growth is possible.

Theorem 3.3 *Suppose we have computed T block steps of $LU\_CRTP(A, k, \tau)$ factorization by using Algorithm 3, where $A \in \mathbb{R}^{m \times n}$ and $K = Tk$. We obtain the following factorization*

$$\begin{aligned} P_r A P_c &= L_K U_K \\ &= \begin{pmatrix} I & & & & \\ L_{21} & I & & & \\ \vdots & \vdots & \ddots & & \\ L_{T1} & L_{T2} & \dots & I & \\ L_{T+1,1} & L_{T+1,2} & \dots & L_{T+1,T} & I \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & \dots & U_{1T} & U_{1,T+1} \\ & U_{22} & \dots & U_{2T} & U_{2,T+1} \\ & & \ddots & \vdots & \vdots \\ & & & U_{TT} & U_{T,T+1} \\ & & & & U_{T+1,T+1} \end{pmatrix} \end{aligned} \tag{34}$$

where $L_{i+1,j}$ and U_{ij} are $k \times k$ for $1 \leq i, j \leq T$, and $U_{T+1,T+1}$ is $(m-Tk) \times (n-Tk)$. The following properties are satisfied:

$$\rho_l(L_{i+1,j}) \leq F_{TP}, \tag{35}$$

$$||U_K||_{max} \leq \min\left((1 + F_{TP}\sqrt{k})^{K/k} ||A||_{max}, q_2(m,k) q(m,n,k)^{K/k-1} \sigma_K(A)\right), \tag{36}$$

$$\frac{1}{\prod_{v=0}^{t-2} q(m-vk, n-vk, k)} \leq \frac{\sigma_{(t-1)k+i}(A)}{\sigma_i(U_{tt})} \leq q(m-(t-1)k, n-(t-1)k, k), \tag{37}$$

$$1 \leq \frac{\sigma_j(U_{T+1,T+1})}{\sigma_{K+j}(A)} \leq \prod_{v=0}^{K/k-1} q(m-vk, n-vk, k), \tag{38}$$

for any $1 \leq l \leq k$, $1 \leq i \leq k$, $1 \leq t \leq T$, and $1 \leq j \leq \min(m,n) - K$. Here F_{TP} is the bound obtained from QR with tournament pivoting as given in Corollary 2.4, $q_2(m,k) = \sqrt{1 + F_{TP}^2(m-k)}$, and $q(m,n,k) = \sqrt{(1 + F_{TP}^2(n-k))\,(1 + F_{TP}^2(m-k))}$.

Proof 3.4 *We consider the first and second step of the block factorization, written as:*

$$P_{r1}AP_{c1} = \begin{pmatrix} I & \\ L_{2:T+1,1} & I \end{pmatrix} \begin{pmatrix} U_{11} & U_{1,2:T+1} \\ & S(U_{11}) \end{pmatrix}, \quad (39)$$

$$P_{r2}S(U_{11})P_{c2} = \begin{pmatrix} I & \\ L_{3:T+1,2} & I \end{pmatrix} \begin{pmatrix} U_{22} & U_{2,3:T+1} \\ & S(U_{22}) \end{pmatrix}, \quad (40)$$

where $L_{2:T+1,1}$ is the first block column formed by $[L_{21}; ...; L_{T+1,1}]$ from equation (34). Similar notation is used for the other block columns of L or block rows of U. We bound first the singular values of the obtained factorization. By applying Theorem 3.1 we have:

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(U_{11})}, \frac{\sigma_j(S(U_{11}))}{\sigma_{k+j}(A)} \leq q(m,n,k), \quad (41)$$
$$\text{for any } 1 \leq i \leq k,\ 1 \leq j \leq \min(m,n) - k$$

$$1 \leq \frac{\sigma_i(S(U_{11}))}{\sigma_i(U_{22})}, \frac{\sigma_j(S(U_{22}))}{\sigma_{k+j}(S(U_{11}))} \leq q(m-k,n-k,k), \quad (42)$$
$$\text{for any } 1 \leq i \leq k,\ 1 \leq j \leq \min(m,n) - 2k.$$

We have that

$$\frac{1}{q(m,n,k)} \leq \frac{\sigma_{k+i}(A)}{\sigma_i(S(U_{11}))} \frac{\sigma_i(S(U_{11}))}{\sigma_i(U_{22})} \leq q(m-k,n-k,k), \text{ for any } 1 \leq i \leq k,$$

and we obtain

$$\frac{1}{q(m,n,k)} \leq \frac{\sigma_{k+i}(A)}{\sigma_i(U_{22})} \leq q(m-k,n-k,k), \text{ for any } 1 \leq i \leq k. \quad (43)$$

We also have

$$1 \leq \frac{\sigma_j(S(U_{22}))}{\sigma_{k+j}(S(U_{11}))} \frac{\sigma_{k+j}(S(U_{11}))}{\sigma_{2k+j}(A)} \leq q(m,n,k)q(m-k,n-k,k),$$

for any $1 \leq j \leq \min(m,n) - 2k$, and we obtain

$$1 \leq \frac{\sigma_j(S(U_{22}))}{\sigma_{2k+j}(A)} \leq q(m,n,k)q(m-k,n-k,k). \quad (44)$$

We consider the steps t and $t+1$ of the factorization,

$$P_{rt}S(U_{t-1,t-1})P_{ct} = \begin{pmatrix} I & \\ L_{t+1:T+1,s} & I \end{pmatrix} \begin{pmatrix} U_{tt} & U_{t,t+1:T+1} \\ & S(U_{tt}) \end{pmatrix}, \quad (45)$$

$$P_{r,t+1}S(U_{tt})P_{c,t+1} = \begin{pmatrix} I & \\ L_{t+2:T+1,t+1} & I \end{pmatrix} \begin{pmatrix} U_{t+1,t+1} & U_{t+1,t+2:T+1} \\ & S(U_{t+1,t+1}) \end{pmatrix}, \quad (46)$$

We suppose that the bounds on singular values at the t-th step of the factorization satisfy the following relations,

$$\frac{1}{\prod_{v=0}^{t-2} q(m-vk,n-vk,k)} \leq \frac{\sigma_{(t-1)k+i}(A)}{\sigma_i(U_{tt})} \leq q(m-(t-1)k,n-(t-1)k,k),$$

$$1 \leq \frac{\sigma_j(S(U_{tt}))}{\sigma_{tk+j}(A)} \leq \prod_{v=0}^{t-1} q(m-vk,n-vk,k),$$

for any $1 \leq i \leq k$ *and* $1 \leq j \leq \min(m,n) - tk$. *In addition by applying Theorem 3.1 to the factorization obtained at the* $(t+1)$*-th step from equation* (46), *we have:*

$$1 \leq \frac{\sigma_i(S(U_{tt}))}{\sigma_i(U_{t+1,t+1})}, \frac{\sigma_j(S(U_{t+1,t+1}))}{\sigma_{k+j}(S(U_{tt}))} \leq q(m-tk, n-tk, k), \quad \text{for any } 1 \leq i \leq k,\ 1 \leq j \leq \min(m,n) - (t+1)k. \tag{47}$$

We have that,

$$\frac{1}{\prod_{v=0}^{t-1} q(m-vk, n-vk, k)} \leq \frac{\sigma_{tk+i}(A)}{\sigma_i(S(U_{tt}))} \frac{\sigma_i(S(U_{tt}))}{\sigma_i(U_{t+1,t+1})} \leq q(m-tk, n-tk, k) \tag{48}$$

$$1 \leq \frac{\sigma_j(S(U_{t+1,t+1}))}{\sigma_{k+j}(S(U_{tt}))} \frac{\sigma_{k+j}(S(U_{tt}))}{\sigma_{(t+1)k+j}(A)} \leq q(m-tk, n-tk, k) \prod_{v=0}^{t-1} q(m-vk, n-vk, k) \tag{49}$$

for any $1 \leq i \leq k$ *and* $1 \leq j \leq min(m,n) - (t+1)k$. *We obtain the following bounds for the* $(t+1)$ *step of block factorization,*

$$\frac{1}{\prod_{v=0}^{t-1} q(m-vk, n-vk, k)} \leq \frac{\sigma_{tk+i}(A)}{\sigma_i(U_{t+1,t+1})} \leq q(m-tk, n-tk, k) \tag{50}$$

$$1 \leq \frac{\sigma_j(S(U_{t+1,t+1}))}{\sigma_{(t+1)k+j}(A)} \leq \prod_{v=0}^{t} q(m-vk, n-vk, k) \tag{51}$$

for any $1 \leq i \leq k$ *and* $1 \leq j \leq min(m,n) - (t+1)k$. *This proves by induction the bounds from equations* (37) *and* (38).

The bound on element growth in L_K *from equation* (35) *follows from equation* (29). *The part of the bound from equation* (36) *that bounds* $||U_K||_{max}$ *with respect to* $||A||_{max}$ *can be easily obtained, and it has similarities with the bound obtained from* LU *with panel rank revealing factorization, see section 2.2.1 in [29]. The part of the bound of* $||U_K||_{max}$ *with respect to the singular values of* A *is obtained as following. After the first iteration, by using the second part of the bound from equation* (30), *we have that* $||S(U_{11})||_{max} \leq F_{TP} q_2(m,k)\sigma_k(A)$. *After the second iteration, by using the second part of the bound from equation* (30) *and equation* (41), *we obtain*

$$||S(U_{22})||_{max} \leq F_{TP} q_2(m-k,k)\sigma_k(S(U_{11})) \leq F_{TP} q_2(m-k,k) q(m,n,k)\sigma_{2k}(A).$$

After the third iteration, by using the second part of the bound from equation (30), *we have that* $||S(U_{33})||_{max} \leq F_{TP} q_2(m-2k,k)\sigma_k(S(U_{22}))$. *By using equation* (44), *we obtain*

$$||S(U_{33})||_{max} \leq F_{TP} q_2(m-2k,k) q(m,n,k) q(m-k,n-k,k)\sigma_{3k}(A).$$

The bound from equation (36) *follows by induction. Note that a tighter bound can be obtained for each row block of* U_K, *however for simplicity we do not give it here.*

Algorithm 3 presents the LU_CRTP(A,k,τ) factorization, which can be used when the rank of the low rank approximation needs to be determined. It can be easily seen from the bounds obtained in Theorem 3.1 and Theorem 3.3 that the R factor obtained from the QR factorization of every block of columns provides a slightly better estimation of the singular values of A than the diagonal blocks of U_K. This is why the algorithm uses the R factor to approximate the singular values of A and determine the rank K of the approximation.

Algorithm 3 LU_CRTP(A, k, τ): rank-K truncated LU factorization with column/row tournament pivoting of a matrix A, using tolerance τ to identify singular values large enough to keep in the low rank approximation

1: **Input** $A \in \mathbb{R}^{m\times n}$, k, tolerance τ
2: **Output** permutation matrices P_r, P_c, rank K, truncated factorization $B = \tilde{L}_K \tilde{U}_K$ such that $\tilde{L}_K \in \mathbb{R}^{m\times K}$, $\tilde{U}_K \in \mathbb{R}^{K\times n}$

$$\tilde{\sigma}_{K-k}(A) \geq \tau > \tilde{\sigma}_K(A), \text{ where } \tilde{\sigma}_k(A) \text{ is K-th singular value of A approximated by the algorithm,}$$

$$P_r A P_c = L_K U_K = \begin{pmatrix} I & & & \\ L_{21} & I & & \\ \vdots & \vdots & \ddots & \\ L_{T1} & L_{T2} & \dots & I \\ L_{T+1,1} & L_{T+1,2} & \dots & L_{T+1,T} \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & \dots & U_{1T} & U_{1,T+1} \\ & U_{22} & \dots & U_{2T} & U_{2,T+1} \\ & & \ddots & \vdots & \vdots \\ & & & U_{TT} & U_{T,T+1} \end{pmatrix},$$

$$\tilde{L}_K = L_K(:, 1:K) \quad \tilde{U}_K = (1:K, :)$$

Note that $U_{T,T+1}$ is not updated during the last iteration.
3: $\bar{A} = A$
4: **for** $T = 1$ to n/k **do**
5: $j = (T-1)k + 1$, $K = j + k - 1$
6: Determine row/column permutations by using Algorithm 2,
$[P_{r_k}, P_{c_k}, L_k, U_k, R_k] \leftarrow LU\_CRTP(\bar{A}(j:m, j:n), k)$
7: $P_{r_T} = \begin{pmatrix} I & \\ & P_{r_k} \end{pmatrix}$, $P_{c_T} = \begin{pmatrix} I & \\ & P_{c_k} \end{pmatrix}$, where I is $(j-1)\times(j-1)$
8: $\bar{A} = P_r A P_c$, $L_K = P_r L_K P_c$, $U_K = P_r U_K P_c$, $P_r = P_{r_T} P_r$, $P_c = P_c P_{c_T}$
9: $U_K(j:K, j:n) = U_k$, $L_K(j:m, j:K) = L_k$
10: **for** $i = 1$ to k **do** $\tilde{\sigma}_{j+i-1}(A) = \sigma_i(R_k)$ **endfor**
11: **if** $\tilde{\sigma}_K(A) < \tau$ **then**
12: Return $\tilde{L}_K = L_K(:, 1:K), \tilde{U}_K = (1:K, :), K, P_r, P_c$
13: **else**
14: Update the trailing matrix,
$\bar{A}(K+1:m, K+1:n) = \bar{A}(K+1:m, K+1:n) - L_k U_k$.
15: **end if**
16: **end for**

3.3 A less expensive LU factorization with column tournament pivoting

In this section we present a less expensive LU factorization with column and row permutations which only satisfies some of the bounds from Theorem 3.1. We present only one step of the block factorization, in which the desired rank is k, the extension to a larger rank K is straightforward.

We refer to this factorization as LU_CTP, LU with column tournament pivoting. Given $A \in \mathbb{R}^{m\times n}$, the factorization selects k columns by using QR factorization with tournament pivoting on the matrix A. The factorization obtained is the following

$$AP_c = \begin{pmatrix} A_{11}^c & A_{12}^c \\ A_{21}^c & A_{22}^c \end{pmatrix} = Q \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix} = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix}$$

where $A_{11}^c \in \mathbb{R}^{k\times k}$, $Q \in \mathbb{R}^{m\times m}$, $Q_{11}, R_{11} \in \mathbb{R}^{k\times k}$. Note that the column permutation is the same as the one used in $LU\_CRTP(A, k)$. However the row permutation is obtained by computing

LU with partial pivoting of the first k columns of AP_c. To reduce communication, LU with tournament pivoting can be used to select the k rows [22], or when necessary for more stability LU with tournament pivoting on the first k columns of Q [13],

$$P_r \begin{pmatrix} A_{11}^c \\ A_{21}^c \end{pmatrix} = \begin{pmatrix} L_{11} \\ L_{21} \end{pmatrix} U_{11},$$

where L_{11}, U_{11} are of dimension $k \times k$. The obtained LU factorization is

$$P_r A P_c = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & \\ L_{21} & I \end{pmatrix} = \begin{pmatrix} U_{11} & U_{12} \\ & S(\bar{A}_{11}) \end{pmatrix},$$

where $S(\bar{A}_{11}) = \bar{A}_{22} - L_{21}U_{12}$. Note that this factorization does not lower bound the singular values of Q_{11}, and this prevents us from bounding the singular values of $\bar{A}_{11}$ and $S(\bar{A}_{11})$ with respect to the singular values of A. However, as we will see in the experimental results, in practice this factorization approximates well the singular values of A.

4 Complexity analysis of QR factorization with tournament pivoting

The selection of k columns by using QR factorization with tournament pivoting dominates the cost of the rank-k approximation factorization from Algorithm 2. We analyze in this section the cost of QR factorization with tournament pivoting for matrices with arbitrary sparsity structure, in terms of fill in the factors Q and R obtained during tournament, floating point operations, as well as interprocessor communication for the parallel version of the algorithm. Given the assumptions we make in our analysis, we expect that these bounds are loose. We then consider matrices whose column intersection graphs have small separators and obtain tighter bounds for the fill in the factors Q and R. In both cases, our analysis is based on a column permutation of the matrix which allows us to bound the fill.

Analyzing the cost of the low rank approximation factorization when the rank K needs to be determined, Algorithm 3, is a more difficult problem that we do not address in this paper. Tournament pivoting could potentially select the most dense k columns, and predicting an upper bound on the fill that occurs in the Schur complement can be very pessimistic.

4.1 Matrices with arbitrary sparsity structure

We discuss now the case of an arbitrary sparse matrix $A \in \mathbb{R}^{m \times n}$. Let d_i be the number of nonzeros in column i of A, $nnz(A) = \sum_{i=1}^{n} d_i$. We permute the matrix columns such that $d_1 \leq \ldots \leq d_n$.

Consider QR_TP with a flat tree. As explained in section 2.2, the matrix A is partitioned into n/k blocks of columns as $A = [A_{00}, \ldots, A_{n/k,0}]$. At the first step of tournament pivoting, the matrix $A_{01} = [A_{00}, A_{10}]$ is formed, and k columns are selected by computing the QR factorization of A_{01}, followed by a rank revealing factorization of the R factor. At the subsequent steps, a new matrix A_{ij} is formed with the previously selected k columns and a block A_{i0} of unvisited columns of A. From this matrix, k new columns are selected by using QR factorization followed by rank revealing factorization of the R factor. Given that k is small, we ignore in the following the cost of the rank revealing factorization of the R factors.

At the first step of reduction, the matrix A_{01} formed by the first $2k$ columns of A has at most $\sum_{i=1}^{2k} d_i$ rows (with equality when the sets of column indices are disjoint and each

row has only one nonzero). By considering this matrix dense, $nnz(A_{01}) \leq 2k \sum_{i=1}^{2k} d_i$ and the number of flops required to compute its QR factorization is at most $8k^2 \sum_{i=1}^{2k} d_i$. The selected k columns have at most $\sum_{i=k+1}^{2k} d_i$ nonzeros and form a matrix with at most $\sum_{i=k+1}^{2k} d_i$ rows. The matrix formed at the second step of tournament pivoting has at most $\sum_{i=k+1}^{3k} d_i$ rows, at most $2k \sum_{i=k+1}^{3k} d_i$ nonzeros, and its QR factorization requires at most $8k^2 \sum_{i=k+1}^{3k} d_i$. These bounds are asymptotically attainable if A has the following nonzero structure: the first $d_1 - 1$ rows have a nonzero only in the first column, while the d_1-th row has nonzeros in the first and the second column. The following $d_1 - 2$ rows have a nonzero in the second column, while the $(d_1 + d_2 - 1)$-th row has nonzeros in the second and the third column. The nonzero structure of the following rows follows the same pattern.

We consider the following costs associated with QR_TP: $nnz_{max}(QR\_TP_{FT})$ refers to the maximum number of nonzeros of the Q and R factors over all the QR factorizations computed during tournament pivoting, $nnz_{total}(QR\_TP_{FT})$ refers to the sum of the number of nonzeros of all the Q and R factors used during tournament pivoting, and $flops(QR\_TP_{FT})$ refers to the total number of flops computed. Note that $nnz_{max}(QR\_TP_{FT})$ reflects the memory needs of QR_TP, since once k columns are selected, the corresponding factors Q and R can be discarded before proceeding to the following step of tournament pivoting. We obtain,

$$
\begin{aligned}
nnz_{max}(QR\_TP_{FT}) &\leq 4d_n k^2 && (52)\\
nnz_{total}(QR\_TP_{FT}) &\leq 2k \left(\sum_{i=1}^{2k} d_i + \sum_{i=k+1}^{3k} d_i + \ldots + \sum_{i=n-2k+1}^{n} d_i \right) \leq \\
&\leq 4k \sum_{i=1}^{n} d_i = 4nnz(A)k, && (53)\\
flops(QR\_TP_{FT}) &\leq 16nnz(A)k^2, && (54)
\end{aligned}
$$

We discuss now the case when QR_TP is executed in parallel by using a binary tree of depth $\log(n/k)$, where $2k \leq n/P$. We consider a column cyclic distribution of the matrix on P processors, and we assume that n/P columns with a total of $nnz(A)/P$ nonzeros are stored on each processor. We also make the reasonable assumption that k is small enough such that $\sum_{i=n-2k+1}^{n} d_i \leq nnz(A)/P$. At each step of the reduction, the matrix A_{ij} has $2k$ columns and at most $\sum_{i=n-2k+1}^{n} d_i$ rows, and each processor executes $\log(n/k)$ QR factorizations of such matrices. The cost of QR_TP per processor is

$$
\begin{aligned}
nnz_{max}(QR\_TP_{BT}) &\leq 2k \sum_{i=n-2k+1}^{n} d_i \leq 2\frac{nnz(A)}{P}k && (55)\\
nnz_{total}(QR\_TP_{BT}) &\leq 2k(\sum_{i=n-2k+1}^{n} d_i) \log \frac{n}{k} \leq 2\frac{nnz(A)}{P}k \log \frac{n}{k}, && (56)\\
flops(QR\_TP_{BT}) &\leq 8k^2(\sum_{i=n-2k+1}^{n} d_i) \log \frac{n}{k} \leq 8\frac{nnz(A)}{P}k^2 \log \frac{n}{k}. && (57)
\end{aligned}
$$

Given our assumption that the matrices used during tournament pivoting have only one nonzero per row, these bounds are loose.

In terms of interprocessor communication, it can be easily seen that during tournament pivoting, the processors exchange $\log P$ messages. Each messages has at most $\sum_{i=n-2k+1}^{n} d_i$ words, for a total of $(\sum_{i=n-2k+1}^{n} d_i) \log P \leq 2kd_n \log P$ words communicated on the critical path of parallel QR_TP.

4.2 Graphs with small separators

Given a matrix $A \in \mathbb{R}^{m\times n}$, the structure of its Q and R factors can be modeled by using the column intersection graph $G_\cap(A)$. We consider that at each iteration of the QR factorization, one Householder reflection is used to annihilate the elements below the diagonal. The Householder matrix H which stores the Householder vectors can be used to implicitly represent Q. Its structure can also be modeled by $G_\cap(A)$. The graph $G_\cap(A)$ is the undirected graph of A^TA (we ignore numerical cancellations), has n vertices (one for each column of A), and an edge between two vertices if the corresponding columns of A have a nonzero in a same row. The filled column intersection graph of A [17], $G^+_\cap(A)$, is the graph of the Cholesky factor L of A^TA and hence it is also the graph of the R factor of the QR factorization. Both the structure of Q and R can be expressed in terms of paths in the column elimination tree of A, $T_\cap(A)$, which is a depth-first spanning tree of $G^+_\cap(A)$. For simplicity, we consider that $G_\cap(A)$ is connected. The nonzero column indices of a row i of Q correspond to vertices that belong to the path in $T_\cap(A)$ going from the vertex corresponding to the first nonzero in row i of A to the root of the tree [18]. Similar definition holds for the last $m-n$ rows of H.

We consider here the case when the column intersection graph belongs to a class S of graphs with small separators, that are closed under the subgraph relation (that is if G is in S and G_1 is a subgraph of G, then G_1 is also in S). We focus on the class S of graphs that are n^λ separable, $\lambda < 1$. These graphs have a separator S with cn^λ vertices, $c > 0$, whose removal disconnects the graph into two graphs A and B, each with less than $2/3n$ vertices. There are many example of graphs with good separators. For example, structured grids in d dimensions are $n^{d-1/d}$-separable, planar graphs are $n^{1/2}$ separable. For more detailed discussions see e.g. [30].

We recall first results established in [19] on the number of nonzeros in the factors R, Q, as well as the Householder vectors H used during the QR factorization of A. These results are obtained by using a reordering of the matrix A based on nested dissection [16], which partitions the graph into two disjoint subgraphs and a separator, and continues recursively the same procedure on the two subgraphs. In the analysis, we consider that c and λ are fixed. We focus first on graphs which are $\sqrt{n}$-separable.

Theorem 4.1 (Theorem 1 from [19]) *Let A be an $m\times n$ matrix of full column rank, such that $G_\cap(A)$ is a member of a $\sqrt{n}$-separable class of graphs. Then there exists a column permutation P such that the matrices Q, R, and H in the thin QR factorization of AP satisfy the following bounds: $nnz(R) = O(n\log n)$, $nnz(Q) = O(m\sqrt{n})$, $nnz(H) = O(n\log n + (m-n)\sqrt{n})$.*

These results are based on the fact that every row of Q has at most $O(\sqrt{n})$ nonzeros, the first n rows of H are a subset of the structure of the rows of R, while the last $m-n$ rows of H have at most $O(\sqrt{n})$ nonzeros each.

Let $A \in \mathbb{R}^{m\times n}$ be a matrix whose column intersection graph $G_\cap(A)$ belongs to the class of $\sqrt{n}$-separable graphs, closed under the subgraph relation. As in the case of arbitrary matrices, we let d_i be the number of nonzeros in column i of A and we consider that A was permuted such that $d_1 \le \ldots \le d_n$. We make the same assumption as before that $kd_n \le nnz(A)/P \le nd_n/P$. In the case of QR with tournament pivoting based on a flat tree, the matrix formed at the first step of tournament A_{01} has $2k$ columns of A and has at most $\sum_{i=1}^{2k} d_i$ rows that have nonzero elements. Given that we consider a class of graphs closed under the subgraph relation, the graph of a submatrix A_{ij} formed by $2k$ columns, $G(A_{ij})$, has $2k$ vertices and satisfies the $(2k)^{1/2}$ separator theorem. We obtain that at the first step of tournament pivoting, the QR factorization of the current matrix A_{01} leads to factors that have $nnz(R) = O(k\log k)$, $nnz(Q) \le O(\sqrt{k}\sum_{i=1}^{2k} d_i)$, and $nnz(H) \le O(k\log k + (\sum_{i=1}^{2k} d_i - k)\sqrt{k})$. Since each row of H has at most $O(\sqrt{k})$ nonzeros,

computing the QR factorization of A_{01} costs at most $O(k^{3/2}\sum_{i=1}^{2k} d_i)$ flops. By summing over the n/k QR factorizations used during tournament pivoting and by using the same approach as for matrices with arbitrary sparsity structure, the cost of QR_TP is

$$\begin{aligned} nnz_{max}(QR\_TP_{FT}) &\leq O(d_n k^{3/2}), & (58)\\ nnz_{total}(QR\_TP_{FT}) &\leq O(nnz(A)\sqrt{k}), & (59)\\ flops(QR\_TP_{FT}) &\leq O(nnz(A)k^{3/2}). & (60) \end{aligned}$$

These bounds are smaller than the bounds for matrices with arbitrary sparsity structure from (52), (53), and (54) by a factor of $O(\sqrt{k})$. We discuss now the case when QR_TP is executed in parallel using a binary reduction tree as before. The reduction tree used during tournament pivoting has depth $\log(n/k)$, and each processor executes $\log(n/k)$ QR factorizations of matrices A_{ij} with $2k$ columns and at most $\sum_{i=n-2k+1}^{n} d_i \leq 2kd_n$ rows that have at least one nonzero. Each such QR factorization can be computed in at most $O(k^{3/2}\sum_{i=n-2k+1}^{n} d_i)$ flops. The cost of QR_TP per processor is

$$\begin{aligned} nnz_{max}(QR\_TP_{BT}) &\leq O(\sqrt{k}\sum_{i=n-2k+1}^{n} d_i) \leq O(d_n k^{3/2}) \leq O(\frac{nnz(A)}{P}\sqrt{k}), & (61)\\ nnz_{total}(QR\_TP_{BT}) &\leq O(\sqrt{k}(\sum_{i=n-2k+1}^{n} d_i)\log\frac{n}{k}) \leq O(\frac{nnz(A)}{P}\sqrt{k}\log\frac{n}{k}), & (62)\\ flops(QR\_TP_{BT}) &\leq O(k^{3/2}(\sum_{i=n-2k+1}^{n} d_i)\log\frac{n}{k}) \leq O(\frac{nnz(A)}{P}k^{3/2}\log\frac{n}{k}). & (63) \end{aligned}$$

These costs are $O(\sqrt{k})$ smaller than the corresponding costs from (55), (56), and (57) obtained for matrices with arbitrary sparsity structure.

By using the same reasoning, these results can be extended to graphs with larger separators. Consider that $G_\cap(A)$ belongs to the class of n^λ-separable graphs, with $1/2 < \lambda < 1$, closed under the subgraph relation. Nested dissection leads to the following bounds of the factors obtained from QR factorization [19], $nnz(R) = O(n^{2\lambda})$, $nnz(Q) = O(mn^\lambda)$, $nnz(H) = O(n^{2\lambda}+(m-n)n^\lambda)$. A path in the column elimination tree $G_\cap(A)$ going from its leaves to its root is formed by the vertices of a separator obtained at each level of nested dissection, and hence its length is $O(n^\lambda)$. We deduce that the number of nonzeros in each row of Q and in the last $m-n$ rows of H is at most $O(n^\lambda)$. We obtain the following bounds for QR_TP:

$$\begin{aligned} nnz_{max}(QR\_TP_{FT}) &= nnz_{max}(QR\_TP_{BT}) \leq O(d_n k^{\lambda+1}) \leq O(\frac{nnz(A)}{P}k^\lambda) & (64)\\ nnz_{total}(QR\_TP_{FT}) &\leq O(nnz(A)k^\lambda) & (65)\\ flops(QR\_TP_{FT}) &\leq O(nnz(A)k^{\lambda+1}) & (66)\\ nnz_{total}(QR\_TP_{BT}) &\leq O(\frac{nnz(A)}{P}k^\lambda\log\frac{n}{k}) & (67)\\ flops(QR\_TP_{BT}) &\leq O(\frac{nnz(A)}{P}k^{\lambda+1}\log\frac{n}{k}) & (68) \end{aligned}$$

We note again that these bounds are smaller than their counterparts in section 4.1 by factors $O(k^{1-\lambda})$.

5 Experimental results

In this section we present experimental results which show that LU_CRTP provides a good low rank approximation in terms of both accuracy and speed. We discuss first the accuracy of LU_CRTP by comparing the approximation of the singular values obtained by our truncated LU factorization with the singular values obtained by SVD. In all the tests, the matrix is first permuted by using COLAMD (column approximate minimum degree) [10] followed by a postorder traversal of its column elimination tree.

5.1 Numerical accuracy

We use a set of 16 challenging matrices which are often used for testing rank revealing algorithms. These matrices are generated in Matlab, they are of size 256×256, and their description can be found in Table 1. A more detailed description can be found in [12]. Experiments are carried out in double precision in Matlab 2015a.

No.	Matrix	Description
1	Baart	Discretization of the 1st kind Fredholm integral equation [25].
2	Break1	Break 1 distribution, matrix with prescribed singular values [3].
3	Break9	Break 9 distribution, matrix with prescribed singular values [3].
4	Deriv2	Computation of second derivative [25].
5	Devil	The devil's stairs, a matrix with gaps in its singular values, see [36] or [12].
6	Exponential	Exponential Distribution, $\sigma_1 = 1$, $\sigma_i = \alpha^{i-1}$ $(i = 2, \ldots, n)$, $\alpha = 10^{-1/11}$ [3].
7	Foxgood	Severely ill-posed test problem of the 1st kind Fredholm integral equation used by Fox and Goodwin [25].
8	Gravity	1D gravity surveying problem [25].
9	Heat	Inverse heat equation [25].
10	Phillips	Phillips test problem [25].
11	Random	Random matrix `A = 2* rand(n) - 1` [23].
12	Shaw	1D image restoration model [25].
13	Spikes	Test problem with a "spiky" solution [25].
14	Stewart	Matrix `A = U` Σ `V`^{T} `+ 0.1` σ_m `* rand(n)`, where σ_m is the smallest nonzero singular value [36].
15	Ursell	Integral equation with no square integrable solution [25].
16	Wing	Test problem with a discontinuous solution [25].

Table 1: Test matrices generated in Matlab.

Figure 1 displays the evolution of singular values obtained by SVD and their approximation obtained by LU_CRTP for two matrices, Exponential and Foxgood. The value of k in LU_CRTP(A, k, τ) from Algorithm 3 is 16. However, the algorithm does not stop at k, but continues the factorization recursively until completion and hence approximates all the singular values of A. The approximated singular values correspond to the diagonal elements of the R factor of each block of k columns obtained from a tournament. In addition, the figure also displays the results obtained by QRCP and those obtained when the column and row permutations are determined by using QRCP instead of tournament pivoting, referred to as LU_CRQRCP. We note that the singular values are well approximated by the three algorithms, and the results obtained by LU_CRQRCP and LU_CRTP are almost superimposed. The usage of tournament pivoting instead of QRCP to select columns and rows in a block LU factorization does not lead

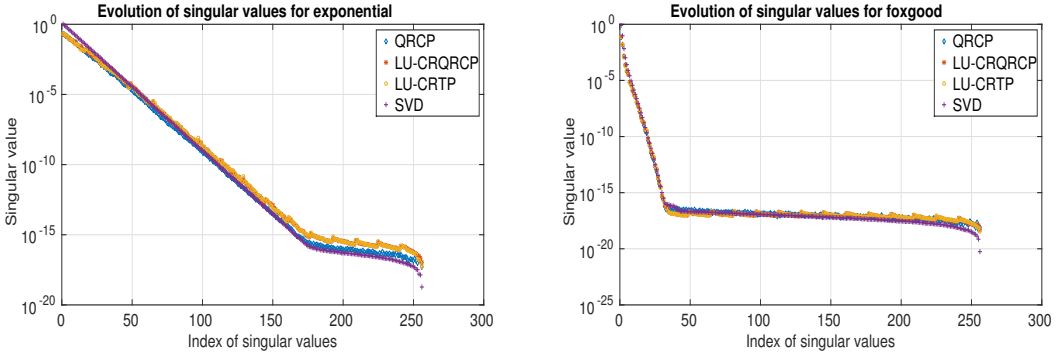


Figure 1: Evolution of the singular values computed by SVD and approximated by QRCP, LU_CRQRCP (LU with column and row pivoting based on QRCP), and LU_CRTP (LU with column and row tournament pivoting).

to loss of accuracy in our experiments. The same behavior is observed for the remaining matrices in the set from Table 1, and the results can be found in the Appendix, Figure 3.

Figure 2 displays the ratios of the approximated singular values with respect to the singular values for the 16 matrices in our set, summarized by the minimum, maximum, and mean values of the ratios $|R_{i,i}|/\sigma_i(A)$. Three different methods are tested, QRCP, LU_CRTP, and LU_CTP. The last method, LU_CTP, corresponds to the cheaper factorization described in section 3.3 in which only the column permutation is selected by using tournament pivoting based on QR, while the row permutation is based on LU with partial pivoting. The bars display the results obtained when the algorithm is truncated at $K = 128$, and the red lines display the results obtained when the factorization runs almost to completion, it is truncated at $K = n - k$. In the results all singular values smaller than machine precision, ϵ, are replaced by ϵ. For the very last k columns, the ratios are slightly larger, and we do not include them in the results, since they are not relevant for a low rank approximation. These results show that the mean is very close to 1. On average, the approximated singular values are very close to the singular values computed by SVD. For the matrices in our set, except devil's stairs, the ratio is between 0.08 and 13.1 for LU_CRTP and between 0.08 and 17.5 for LU_CTP. For the devil's stairs, a more challenging problem for rank revealing factorizations, the maximum ration for LU_CRTP is 27 and the maximum ratio for LU_CTP is 26.

5.2 Performance

We discuss first the performance of our algorithm by comparing the number of nonzeros in the factors of LU_CRTP with respect to the number of nonzeros in the factors of QRCP and LU with partial pivoting. As mentioned earlier, for all the factorizations, the columns of the matrix were first permuted by using COLAMD followed by a postorder traversal of its column elimination tree. The number of nonzeros in the factors gives not only the memory usage of these factorizations, but also a good indicator of their expected performance. We use several larger square sparse matrices obtained from the University of Florida Sparse Matrix Collection [11]. Table 2 displays the name of the matrices, their number of columns/rows (Size), their number of nonzeros (Nnz), and their field of application. Some of the matrix names are abbreviated. The matrix TSOPF_RS corresponds to the matrix TSOPF_RS_B39_C30 in the collection, PARAB_FEM corresponds to PARABOLIC_FEM, while MAC_ECON corresponds to MAC_ECON_FWD500.

Table 3 displays the results obtained for the first 10 matrices from Table 2 when a rank K

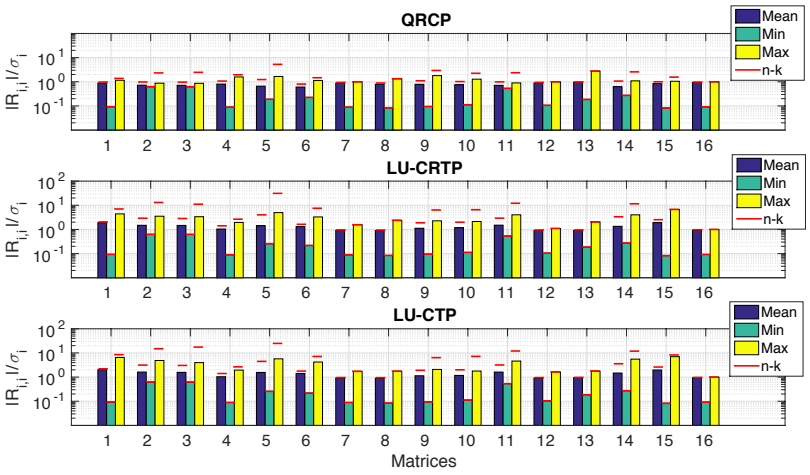


Figure 2: Comparison of approximations of singular values obtained by LU_CRTP, LU_CTP, and QRCP for the matrices described in Table 1. Here $k = 16$ and the factorization is truncated at $K = 128$ (bars) or $K = 240$ (red lines) .

No.	Matrix	Size	Nnz	Problem description
17	ORANI678	2529	90158	Economic
18	GEMAT11	4929	33108	Power network sequence
19	RAEFSKY3	21200	1488768	Computational fluid dynamics
20	WANG3	26064	177168	Semiconductor device
21	ONETONE2	36057	222596	Frequency-domain circuit simulation
22	TSOPF_RS	60098	1079986	Power network
23	RFDEVICE	74104	365580	Semiconductor device
24	NCVXQP3	75000	499964	Optimisation
25	MAC_ECON	206500	1273389	Economic
26	PARAB_FEM	525825	3674625	Computational fluid dynamics
27	ATMOSMODD	1270432	8814880	Fluid dynamics
28	CIRCUIT5M_DC	3523317	14865409	Circuit simulation

Table 2: Sparse matrices from University of Florida collection [11].

approximation is computed, where K varies from 128 to 1024. The initial rank $k = 16$. Matlab is used for these experiments, and there was not enough memory to obtain results for the last matrices in Table 2. The second column, $nnz\ A(:, 1 : K)$, displays the number of nonzeros in the first K columns of A, once it was permuted as explained previously. The fourth column, $\frac{nnz\ QRCP}{nnz\ LU\_CRTP}$, displays the ratio of the number of nonzeros of QRCP with respect to the number of nonzeros of LU_CRTP. The last column displays $\frac{nnz\ LU\_CRTP}{nnz\ LUPP}$, the ratio of the number of nonzeros of LU_CRTP with respect to LU with partial pivoting. For QRCP, we count the number of nonzeros in the first K columns of the Householder vectors H plus the number of nonzeros in the first K rows of the R factor. For LU_CRTP and LUPP we count the number of nonzeros in the first K columns of L and the first K rows of U. For LU_CRTP we ignore the number of nonzeros created during tournament pivoting since the memory requirements are small compared to the memory requirements displayed in Table 3.

Name	$nnz\ A(:, 1:K)$	Rank K	$\frac{nnz\ QRCP}{nnz\ LU\_CRTP}$	$\frac{nnz\ LU\_CRTP}{nnz\ LUPP}$
ORANI678	7901	128	17.87	3.30
	55711	512	6.18	8.85
	71762	1024	4.86	11.01
GEMAT11	1232	128	2.1	2.2
	4895	512	3.3	2.6
	9583	1024	11.5	3.2
RAEFSKY3	7872	128	1.25	2.26
	31248	512	1.07	4.18
	63552	1024	1.06	6.58
WANG3	896	128	3.0	2.1
	3536	512	2.9	2.1
	7120	1024	2.9	1.2
ONETONE2	4328	128	36.0	2.8
	9700	512	73.5	1.1
	17150	1024	108.5	0.3
TSOPF_RS	4027	128	2.57	1.90
	5563	512	0.83	2.41
	7695	1024	0.61	2.13
RFDEVICE	633	128	10.0	1.1
	2255	512	82.6	0.9
	4681	1024	207.2	0.9
NCVXQP3	1263	128	2.87	1.21
	5067	512	3.50	1.01
	10137	1024	3.83	0.53
MAC_ECON	384	128	-	0.34
	1535	512	-	0.19
	5970	1024	-	0.11
PARAB_FEM	896	128	-	0.5
	3584	512	-	0.3
	7168	1024	-	0.2

Table 3: Comparison of number of nonzeros of LU with partial pivoting, QRCP, and LU_CRTP. A dash in the table indicates that there was not enough memory to run QRCP to completion.

We note that for smaller matrices, LU_CRTP leads to a factor of up to 17 times fewer nonzeros than QRCP for ORANI678. Larger improvement with respect to QRCP is observed for ONETONE2 and RFDEVICE, up to a factor of 207. As one can expect, LU_CRTP leads to up to 11 times more nonzeros than LUPP. We were not able to run QRCP for the last two matrices in Table 3 due to memory consumption. We also observe that for the last four matrices, LU_CRTP has fewer nonzeros than LUPP. This means that the columns selected by tournament pivoting generate less fill-in than those selected before the factorization by using COLAMD and postorder traversal of the elimination tree as used in LUPP. This is something that we do not expect to happen in general.

We discuss now the parallel performance of LU_CRTP. There are routines for computing the QRCP factorization of a dense matrix as in LAPACK or ScaLAPACK, or for computing the QR factorization of a sparse matrix, as the multifrontal SPQR software of Tim Davis [9]. However there is no library available for computing in parallel a sparse rank revealing factorization. We present in this paper the performance of LU_CRTP(A, k) (Algorithm 2), which given a rank k, computes a rank-k approximation and produces a CUR factorization as presented in (5). The main step of this algorithm is to select k columns from the matrix A by using tournament

pivoting based on QR. We do not have yet a parallel implementation of LU_CRTP(A, k, τ) which computes all the singular values larger than a parameter τ, as given in Algorithm 3, this remains future work. Table 5 reports runtimes for tournament pivoting based on QR, which selects $k = 256$ columns from the input matrix. We use in these tests the last larger matrices from our set in Table 2. The results are obtained on Edison, a Cray XC30 supercomputer at NERSC, formed by nodes with 2 x 12-core Intel "Ivy Bridge" processors. We run as many MPI processes per node as cores. Tournament pivoting uses a binary tree. At each step the selection of k columns from $2k$ columns is performed by first calling SPQR [9], and then calling the QRCP dGEQP3 routine from Lapack [1] (as implemented in MKL) on the R factor obtained from SPQR. SPQR reorders the matrix by using Metis [28].

Table 4 presents the breakdown of the times required for selecting $k = 256$ columns using $P = 32$ MPI processes. The second column displays the time required for reordering each matrix by using COLAMD followed by a postorder traversal of its column elimination tree. The third column displays the time required for selecting k columns from $2k$ columns by calling SPQR and DGEQP3. The runtimes are shown on three rows, the first row displays the minimum runtime, the second row displays the average runtime, while the last row displays the maximum runtime among processors. These statistics are computed over all calls to SPQR and DGEQP3 performed during tournament pivoting. The fourth column displays the time required for selecting k columns locally on each processor from the matrix of size $m \times n/P$ that it owns. Each processor uses a binary tree based tournament pivoting. As for the previous column, we display first the minimum runtime, then the average runtime, and finally the maximum runtime among processors. The last column displays the time required for selecting the final k columns from the sets of k columns selected locally on each of the P processors. The binary tree used by tournament pivoting among processors has depth $\log P$. Our implementation uses a reduction like operation in which the result is available only on one processor, namely processor 0. The number of processors involved at each level of the reduction is halved. Hence we display only the maximum runtime obtained on processor 0 that owns the result and is the root of the reduction tree.

These results show that selecting k columns from $2k$ columns can be performed efficiently. However, for a same matrix, the time spent in SPQR can vary considerably. For example for MAC_ECOM, SPQR varies between 0.06 seconds and 22.71 seconds, and takes on average 3.26 seconds. This results in load imbalance during tournament pivoting both in the first steps performed locally on each processor and in the last $\log P$ steps that involve communication. Essentially most of the time is spent in SPQR. We note that the time spent in the selection of k columns locally can be further reduced by using a flat tree. Our future work will focus on this aspect as well as on desiging a load balanced tournament pivoting.

Table 5 presents the runtime of binary tree based tournament pivoting when increasing the number of cores from 32 to 1024. For the matrices in our test set, the algorithm is scalable up to 1024 cores. For example, selecting 256 columns from PARAB_FEM requires only 5 secs on 1024 cores. The runtime decreases only slightly when the number of processors is larger than 1024. This is because the number of columns at the leaves approaches $2k$ and most of the time is spent in the reduction tree. However more parallelism is possible by choosing a smaller k, updating the trailing matrix and performing more applications of tournament pivoting until the desired low rank approximation is obtained. In this case, k becomes a tuning parameter.

6 Conclusions and future work

In this paper we have introduced LU_CRTP, a block LU factorization based on column and row permutations for computing a low rank approximation of a sparse matrix. The selection

Matrix	COLAMD +etree	Matrix $m \times 2k$ SPQR	Matrix $m \times 2k$ DGEQP3	Tournament pivoting on each processor SPQR	Tournament pivoting on each processor DGEQP3	Tournament pivoting among processors SPQR	Tournament pivoting among processors DGEQP3
MAC_ECON		0.06	0.01	180.97	0.33		
		3.26	0.02	246.65	0.36		
	0.52	22.71	0.02	361.84	0.39	3.37	0.08
PARAB_FEM		0.20	0.01	54.66	0.94		
		0.24	0.02	79.88	1.03		
	0.56	0.29	0.07	103.49	1.13	1.06	0.07
ATMOSMODD		0.48	0.01	183.36	2.17		
		0.92	0.02	236.69	2.39		
	3.35	4.96	0.02	483.61	2.63	2.50	0.05
CIRCUIT5M_DC		1.36	0.01	623.09	6.26		
		1.59	0.02	664.68	6.85		
	4.18	1.84	0.03	749.20	7.99	7.15	0.08

Table 4: Runtime in seconds of tournament pivoting for selecting $k = 256$ columns on $P = 32$ MPI processes. The time is divided between the time required for reordering (COLAMD + etree), the time required for selecting k columns from n/P columns locally on each processor, and the time required for selecting k columns from the sets of columns selected by each processor (the last $\log P$ steps of tournament pivoting). For each matrix, the third and the fourth columns display on different rows minimum, average, and maximum runtimes obtained among different processors. The last column displays only the maximum runtime obtained among different processors.

Matrix	Number MPI processes 32	64	128	256	512	1024	2048
MAC_ECON	367	183	118	83	57	19	12
PARAB_FEM	106	65	36	22	15	5	4
ATMOSMODD	488	269	163	83	52	29	18
CIRCUIT5M_DC	771	367	196	109	69	44	37

Table 5: Runtime in secs for selecting $k = 256$ columns using tournament pivoting based on QR.

of columns and rows at each step of the block factorization uses tournament pivoting based on QR. The experimental results show that LU_CRTP provides a good trade-off between accuracy and speed. The approximated singular values are on avergage very close to the singular values computed by SVD, and in the worst case, are within a factor of 10. On 1024 cores of a Cray XC30 computer, the algorithm computes a rank-256 approximation in 5 seconds for a matrix of size 525825×525825 (PARAB_FEM).

References

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. W. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, SIAM, Philadelphia, PA, USA, 1999.

[2] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz, *Minimizing communication in numerical linear algebra*, SIAM Journal on Matrix Analysis and Applications, 32 (2011), pp. 866–901.

[3] C. H. Bischof, *A parallel QR factorization algorithm with controlled local pivoting*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 36–57.

[4] P. A. Businger and G. H. Golub, *Linear least squares solutions by Householder transformations*, Numer. Math., 7 (1965), pp. 269–276.

[5] T. F. Chan and P. C. Hansen, *Some applications of the rank revealing QR factorization*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 727–741.

[6] S. Chandrasekaran and I. C. F. Ipsen, *On Rank-Revealing Factorisations*, SIAM. J. Matrix Anal. Appl., 15 (1994), pp. 592–622.

[7] K. L. Clarkson and D. P. Woodruff, *Low rank approximation and regression in input sparsity time*, in Proceedings of the forty-fifth annual ACM symposium on Theory of computing, STOC '13, 2013, pp. 81–90.

[8] J. K. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, vol. I: Theory, SIAM, Philadelphia, 2002.

[9] T. Davis, *Algorithm 915, suitesparseqr: Multifrontal multithreaded rank-revealing sparse qr factorization*, ACM Transactions on Mathematical Software, 38 (2011), pp. 8:1 – 8:22.

[10] T. A. Davis, J. R. Gilbert, S. I. Larimore, and E. G. Ng, *A column approximate minimum degree ordering algorithm*, ACM Transactions on Mathematical Software, 30 (2004), pp. 353–376.

[11] T. A. Davis and Y. Hu, *The university of florida sparse matrix collection*, ACM Transactions on Mathematical Software, 38 (2011), pp. 1 – 25. http://www.cise.ufl.edu/research/sparse/matrices.

[12] J. Demmel, L. Grigori, M. Gu, and H. Xiang, *Communication-avoiding rank-revealing QR decomposition*, SIAM Journal on Matrix Analysis and its Applications, 36 (2015), pp. 55–89.

[13] ——, *LU with tournament pivoting for nearly singular matrices*, tech. report, Inria and UC Berkeley, 2015. in preparation.

[14] J. W. Demmel, L. Grigori, M. Hoemmen, and J. Langou, *Communication-optimal parallel and sequential QR and LU factorizations*, SIAM J. Sci. Comput., 34 (2012), pp. 206–239. short version of technical report UCB/EECS-2008-89 from 2008.

[15] J. W. Demmel, N. J. Higham, and R. Schreiber, *Block LU factorization*, Numerical Linear Algebra with Applications, 2 (1995), pp. 173–190.

[16] A. George, *Nested Dissection of a Regular Finite Element Mesh*, SIAM Journal on Numerical Analysis, 10 (1973), pp. 345 – 363.

[17] A. George and J. W.-H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, N.J., 1981.

[18] A. George, J. W.-H. Liu, and E. G.-Y. Ng, *A data structure for sparse QR and LU factors*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 100–121.

[19] J. R. Gilbert, E. G. Ng, and B. W. Peyton, *Separators and structure prediction in sparse orthogonal factorization*, Linear Algebra and its Applications, 262 (1997).

[20] G. H. Golub, *Numerical methods for solving linear least squares problems*, Numer. Math., 7 (1965), pp. 206–216.

[21] Zamarashkin NL Goreinov SA, Tyrtyshnikov EE, *A theory of pseudoskeleton approximations*, Linear Algebra and Its Applications, 261 (1997), pp. 1–21.

[22] L. Grigori, J. W. Demmel, and H. Xiang, *CALU: A communication optimal LU factorization algorithm*, SIAM Journal on Matrix Analysis and Applications, 32 (2011), pp. 1317–1350.

[23] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.

[24] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217 – 288.

[25] P. C. Hansen, *Regularization tools version 4.1 for matlab 7.3*.

[26] Y. P. Hong and C.-T. Pan, *Rank-revealing QR factorizations and the singular value decomposition*, Math. Comp., 58 (1992), pp. 213–232.

[27] W. M. Kahan, *Numerical linear algebra*, Canad. Math. Bull., 9 (1966), pp. 757 – 801.

[28] G. Karypis and V. Kumar, *A software package for partitioning unstructured graphs, partitioning meshes and computing fill-reducing orderings of sparse matrices - verstion 4.0*, 1998. See http://www-users.cs.umn.edu/karypis/metis.

[29] A. Khabou, J. W. Demmel, L. Grigori, and M. Gu, *Communication avoiding LU factorization with panel rank revealing pivoting*, SIAM Journal on Matrix Analysis and Applications, 34 (2013), pp. 1401 – 1429.

[30] R. L. Lipton and R. E. Tarjan, *A separator theorem for planar graphs*, SIAM J. Appl. Math., 36 (1979), pp. 177 – 189.

[31] M. W. Mahoney, *Randomized algorithms for matrices and data*, Found. Trends Mach. Learn., 3 (2011), pp. 123–224.

[32] L. Miranian and M. Gu, *Strong rank revealing LU factorizations*, Linear Algebra and its Applications, (2003), pp. 1–16.

[33] C.-T. Pan, *On the existence and computation of rank-revealing LU factorizations*, Linear Algebra and its Applications, 316 (2000), pp. 199–222.

[34] Y. Saad, *Numerical Methods for Large Eigenvalue Problems, 2nd ed.*, SIAM, Philadelphia, 2011.

[35] G.W. Stewart, *Four algorithms for the efficient computation of truncated QR approximations to a sparse matrix*, Numer. Math., 83 (1999), pp. 313–323.

[36] G. W. Stewart, *The QLP approximation to the singular value decomposition*, SIAM J. Sci. Comput., 20 (1999), pp. 1336–1348.

7 Appendix

In this section we discuss in more detail the numerical accuracy of LU_CRTP, the block LU factorization with column and row permutations.

Figure 3 displays the ratios of singular values approximated by three different methods to the singular values as computed by SVD for all the matrices in our set from Table 1. The three different methods are QRCP, LU with column and row pivoting based on QRCP (referred to as LU_CRQRCP), and LU with column and row tournament pivoting, LU_CRTP. We note that the results obtained by LU_CRTP are very close to the results obtained by LU_CRQRCP. In other words, the columns and rows selected by tournament pivoting in the block LU factorization lead to results comparable to those obtained by selecting columns and rows using QRCP.

The evolution of the singular values obtained by SVD and their approximations obtained by LU_CRTP and LU_CTP (the less expensive variant described in section 3.3) for different matrices from Table 1 is displayed in figures 4 and 5.

We discuss now the numerical accuracy of the factors L and U produced by LU_CRTP(A, k, τ) (Algorithm 3) on the set of matrices described in Table 2. The algorithm stops after computing an approximation of rank $K = 1024$, and k varies from 16 to 128. The obtained factorization $P_r A P_c = L_K U_K$ is the factorization from equation (34). The results in Table 6 display the growth factor of the factorization defined as $g_W = \frac{max_{i,j,k}|a_{i,j}^{(k)}|}{max_{i,j}|a_{i,j}|}$, where $a_{i,j}^{(k)}$ denotes the entry in position (i, j) of A after k iterations. The table also displays different norms of the factors L_K and U_K and their inverses. The last column displays the backward error of the block LU factorization $\frac{||P_r A P_c - LU||_F}{||A||_F}$, when the factorization is run until the end. The results show that the obtained factorization is very stable, the growth factor g_W is equal to 1, backward error varies from 10^{-16} to 10^{-24}.

Matrix	k	g_w	$\|\|L_K\|\|_1$	$\|\|L_K^{-1}\|\|_1$	$\|\|L_K\|\|_{max}$	$\|\|U_K\|\|_1$	$\|\|U_K^{-1}\|\|_1$	$\|\|U_K\|\|_{max}$	$\|\|A\|\|_{max}$	$\frac{\|\|P_r A P_c - LU\|\|_F}{\|\|A\|\|_F}$
orani678	16	1.00e+00	1.43e+02	1.12e+02	1.49e+00	5.52e+01	1.40e+02	4.09e+00	4.09e+00	2.17e-16
	32	1.00e+00	1.42e+02	1.16e+02	1.01e+00	7.65e+01	1.93e+02	4.09e+00	4.09e+00	1.91e-16
	64	1.00e+00	1.31e+02	1.18e+02	1.00e+00	9.73e+01	2.22e+02	4.09e+00	4.09e+00	1.56e-16
	128	1.00e+00	1.36e+02	1.58e+02	1.00e+00	1.45e+02	2.88e+02	4.09e+00	4.09e+00	1.51e-16
gemat11	16	1.00e+00	5.22e+00	5.32e+00	1.00e+00	6.51e+02	3.76e+00	6.23e+02	6.23e+02	4.67e-17
	32	1.00e+00	5.22e+00	5.32e+00	1.00e+00	6.51e+02	5.84e+00	6.23e+02	6.23e+02	4.77e-17
	64	1.00e+00	5.22e+00	5.32e+00	1.00e+00	6.51e+02	5.84e+00	6.23e+02	6.23e+02	4.67e-17
	128	1.00e+00	5.22e+00	5.32e+00	1.00e+00	6.51e+02	6.01e+00	6.23e+02	6.23e+02	4.64e-17
raefsky3	16	1.00e+00	1.00e+00	1.00e+00	1.00e+00	7.99e+00	1.14e+00	7.99e+00	7.99e+00	6.25e-17
	32	1.00e+00	1.00e+00	1.00e+00	1.00e+00	7.99e+00	1.14e+00	7.99e+00	7.99e+00	6.07e-17
	64	1.00e+00	1.00e+00	1.00e+00	1.00e+00	7.99e+00	1.14e+00	7.99e+00	7.99e+00	4.92e-17
	128	1.00e+00	1.00e+00	1.00e+00	1.00e+00	7.99e+00	1.14e+00	7.99e+00	7.99e+00	3.04e-17
wang3	16	1.00e+00	2.00e+00	2.29e+00	1.00e+00	1.06e+00	1.91e+01	1.69e-01	1.69e-01	3.29e-17
	32	1.00e+00	2.00e+00	2.28e+00	1.00e+00	1.06e+00	1.91e+01	1.69e-01	1.69e-01	3.04e-17
	64	1.00e+00	2.00e+00	2.27e+00	1.00e+00	1.06e+00	1.91e+01	1.69e-01	1.69e-01	2.41e-17
	128	1.00e+00	2.00e+00	2.14e+00	1.00e+00	1.06e+00	1.91e+01	1.69e-01	1.69e-01	2.47e-17
onetone2	16	1.00e+00	4.04e+00	4.04e+00	1.00e+00	2.01e+03	1.50e+00	2.01e+03	2.01e+03	2.13e-17
	32	1.00e+00	4.04e+00	4.04e+00	1.00e+00	2.01e+03	1.50e+00	2.01e+03	2.01e+03	2.13e-17
	64	1.00e+00	4.04e+00	4.04e+00	1.00e+00	2.01e+03	1.50e+00	2.01e+03	2.01e+03	2.13e-17
	128	1.00e+00	4.04e+00	4.04e+00	1.00e+00	2.01e+03	1.50e+00	2.01e+03	2.01e+03	2.12e-17
TSOPF_RS	16	1.00e+00	9.19e+01	9.68e+01	1.00e+00	1.11e+03	1.19e+02	6.46e+02	6.46e+02	7.07e-17
	32	1.00e+00	9.90e+01	9.90e+01	1.00e+00	1.24e+03	1.01e+02	6.46e+02	6.46e+02	5.41e-17
	64	1.00e+00	9.90e+01	9.90e+01	1.00e+00	1.39e+03	1.00e+02	6.46e+02	6.46e+02	4.39e-17
	128	1.00e+00	1.74e+01	1.98e+01	1.00e+00	1.39e+03	3.60e+01	6.46e+02	6.46e+02	4.42e-17
RFdevice	16	1.00e+00	1.00e+00	1.00e+00	1.00e+00	5.71e+08	1.00e+00	1.50e+08	1.50e+08	3.98e-24
	32	1.00e+00	1.00e+00	1.00e+00	1.00e+00	5.71e+08	1.00e+00	1.50e+08	1.50e+08	4.06e-24
	64	1.00e+00	1.00e+00	1.00e+00	1.00e+00	5.71e+08	1.00e+00	1.50e+08	1.50e+08	3.90e-24
	128	1.00e+00	1.00e+00	1.00e+00	1.00e+00	5.71e+08	1.00e+00	1.50e+08	1.50e+08	4.18e-24
ncvxqp3	16	1.00e+00	3.80e+00	3.80e+00	1.00e+00	4.25e+05	2.21e+00	4.25e+05	4.25e+05	1.94e-17
	32	1.00e+00	3.80e+00	3.80e+00	1.00e+00	4.25e+05	2.21e+00	4.25e+05	4.25e+05	1.93e-17
	64	1.00e+00	3.80e+00	3.80e+00	1.00e+00	4.25e+05	2.21e+00	4.25e+05	4.25e+05	1.94e-17
	128	1.00e+00	3.80e+00	3.80e+00	1.00e+00	4.25e+05	2.21e+00	4.25e+05	4.25e+05	1.93e-17
mac_econ	16	1.00e+00	3.17e+00	3.17e+00	1.00e+00	1.45e+05	1.11e+00	1.45e+05	1.45e+05	1.40e-18
	32	1.00e+00	3.17e+00	3.17e+00	1.00e+00	1.45e+05	1.11e+00	1.45e+05	1.45e+05	1.40e-18
	64	1.00e+00	3.17e+00	3.17e+00	1.00e+00	1.45e+05	1.11e+00	1.45e+05	1.45e+05	1.40e-18
	128	1.00e+00	3.17e+00	3.17e+00	1.00e+00	1.45e+05	1.11e+00	1.45e+05	1.45e+05	1.40e-18
parab_fem	16	1.00e+00	2.00e+00	2.00e+00	1.00e+00	1.20e+00	2.50e+00	4.00e-01	4.00e-01	2.07e-18
	32	1.00e+00	2.00e+00	2.00e+00	1.00e+00	1.20e+00	2.50e+00	4.00e-01	4.00e-01	2.08e-18
	64	1.00e+00	2.00e+00	2.00e+00	1.00e+00	1.20e+00	2.50e+00	4.00e-01	4.00e-01	2.08e-18
	128	1.00e+00	2.00e+00	2.00e+00	1.00e+00	1.20e+00	2.50e+00	4.00e-01	4.00e-01	2.07e-18

Table 6: Numerical stability of the LU factorization produced by LU_CRTP. For a given rank k, the table displays the growth factor g_W, and different norms of the obtained factors L and U. It also displays in the last column the backward error of the factorization when it is run to completion.

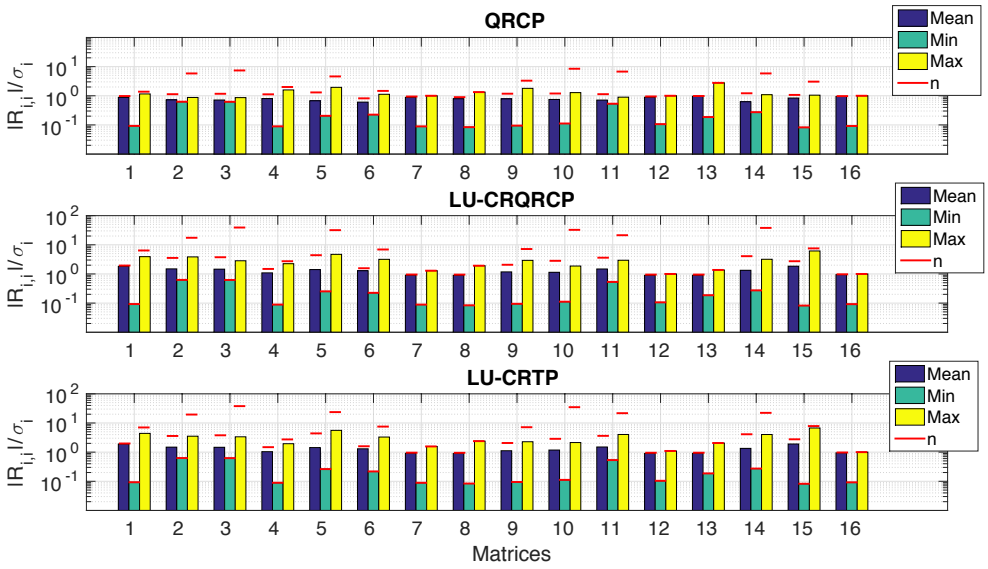


Figure 3: Comparison of approximations of singular values obtained by LU_CRTP, LU_CRQRCP, and QRCP with respect to the singular values computed by SVD. The test matrices are described in Table 1. Here $k = 16$ and the factorization is truncated at $K = 128$ (bars) or it runs to completion (red lines) .

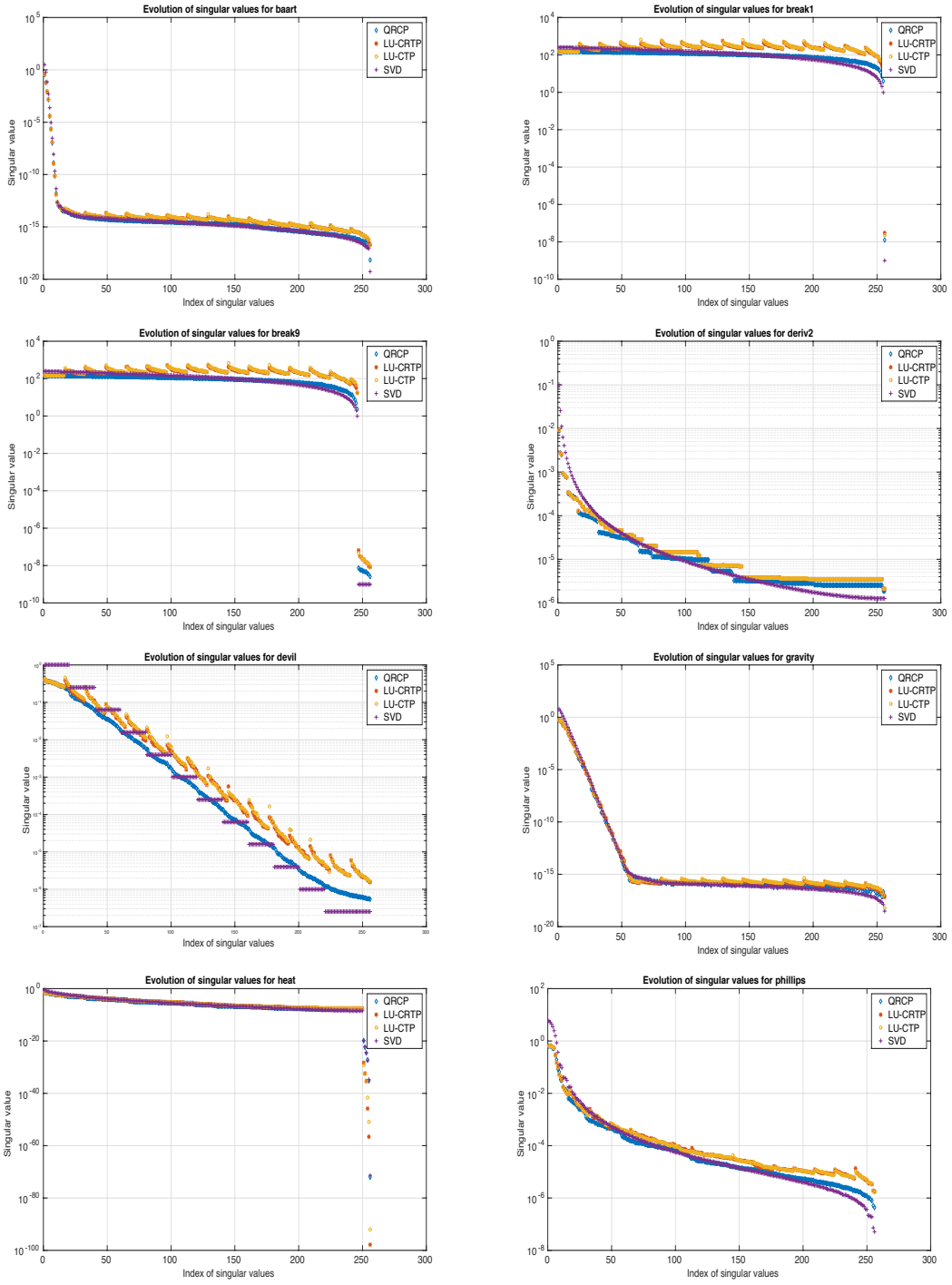


Figure 4: Evolution of the singular values computed by SVD and approximated by QRCP, LU_CRTP (LU with column and row tournament pivoting), and LU_CTP (LU with column tournament pivoting and row pivoting based on partial pivoting).

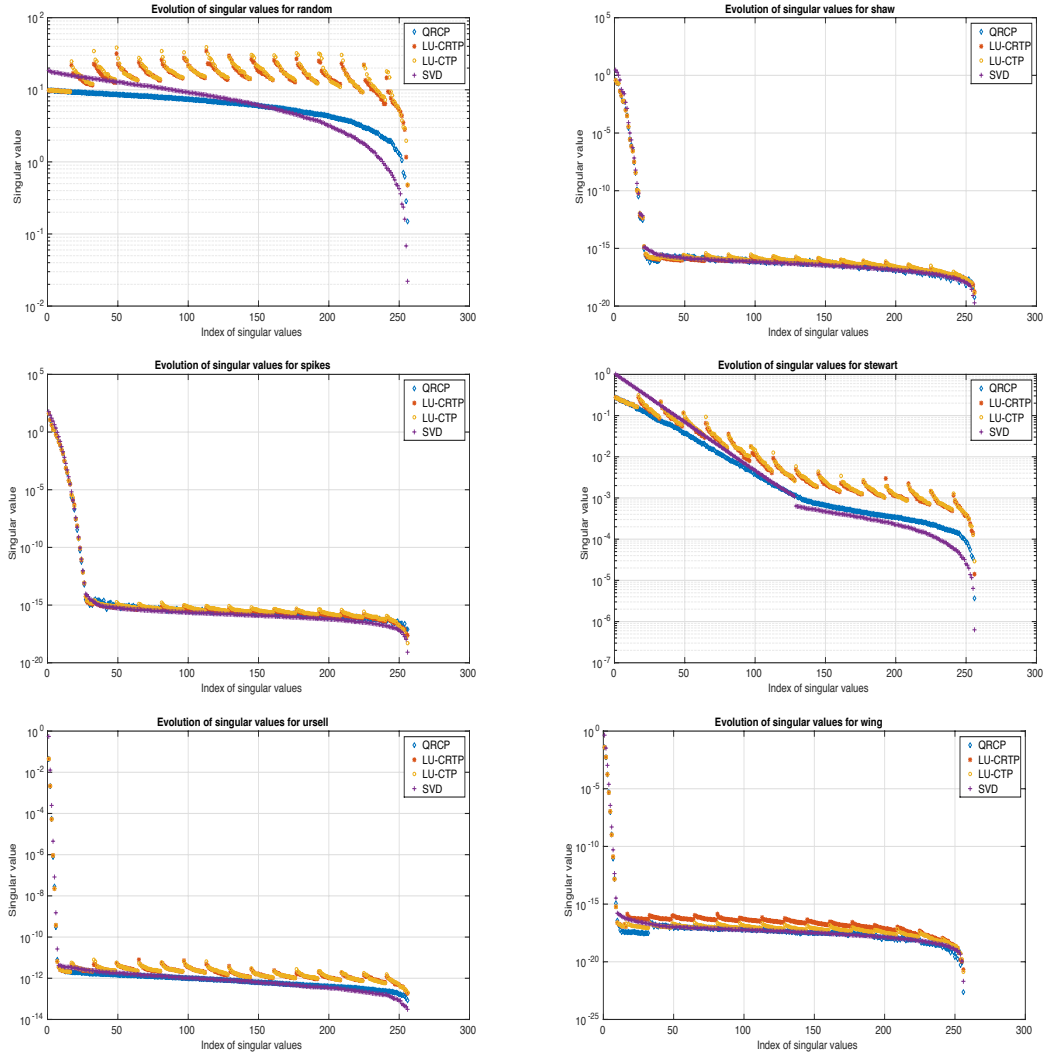
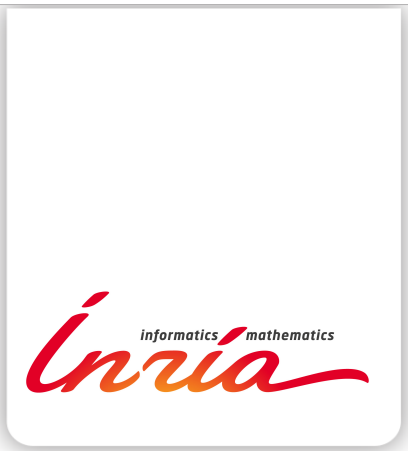


Figure 5: Evolution of the singular values computed by SVD and approximated by QRCP, LU_CRTP (LU with column and row tournament pivoting), and LU_CTP (LU with column tournament pivoting and row pivoting based on partial pivoting).



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399